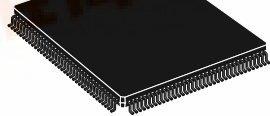




ST10F276

16-bit MCU
with MAC unit, 832 Kbyte Flash memory and 68 Kbyte RAM

Features

- Highly performant 16-bit CPU with DSP functions
 - 31.25ns instruction cycle time at 64MHz max CPU clock
 - Multiply/accumulate unit (MAC) 16 x 16-bit multiplication, 40-bit accumulator
 - Enhanced boolean bit manipulations
 - Single-cycle context switching support
 - On-chip memories
 - 512 Kbyte Flash memory (32-bit fetch)
 - 320 Kbyte extension Flash memory (16-bit fetch)
 - Single voltage Flash memories with erase/program controller and 100K erasing/programming cycles.
 - Up to 16 Mbyte linear address space for code and data (5 Mbytes with CAN or I²C)
 - 2 Kbyte internal RAM (IRAM)
 - 66 Kbyte extension RAM (XRAM)
 - External bus
 - Programmable external bus configuration & characteristics for different address ranges
 - Five programmable chip-select signals
 - Hold-acknowledge bus arbitration support
 - Interrupt
 - 8-channel peripheral event controller for single cycle interrupt driven data transfer
 - 16-priority-level interrupt system with 56 sources, sampling rate down to 15.6ns
 - Timers
 - Two multi-functional general purpose timer units with 5 timers
 - Two 16-channel capture / compare units
- 

PQFP144 (28 x 28 x 3.4mm) LQFP144 (20 x 20 x 1.4mm)
(Plastic Quad Flat Package) (Low Profile Quad Flat Package)
- 4-channel PWM unit + 4-channel XPWM
 - A/D converter
 - 24-channel 10-bit
 - 3 μ s minimum conversion time
 - Serial channels
 - Two synch. / asynch. serial channels
 - Two high-speed synchronous channels
 - One I²C standard interface
 - 2 CAN 2.0B interfaces operating on 1 or 2 CAN busses (64 or 2x32 message, C-CAN version)
 - Fail-safe protection
 - Programmable watchdog timer
 - Oscillator watchdog
 - On-chip bootstrap loader
 - Clock generation
 - On-chip PLL with 4 to 12 MHz oscillator
 - Direct or prescaled clock input
 - Real time clock and 32 kHz on-chip oscillator
 - Up to 111 general purpose I/O lines
 - Individually programmable as input, output or special function
 - Programmable threshold (hysteresis)
 - Idle, power down and stand-by modes
 - Single voltage supply: 5V \pm 10% (embedded regulator for 1.8 V core supply)

Order Codes

| Part Number | Package | Max CPU frequency | lflash | Xflash | RAM | Temperature range (°C) |
|--------------|---------|-------------------|--------|--------|------|------------------------|
| ST10F276Z5Q3 | PQFP144 | 64 MHz | 512KB | 320KB | 68KB | -40/+125 |
| ST10F276Z5T3 | LQFP144 | 40 MHz | 512KB | 320KB | 68KB | -40/+125 |

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 13 |
| 2 | Pin data | 16 |
| 3 | Functional description | 23 |
| 4 | Internal Flash memory | 24 |
| 4.1 | Overview | 24 |
| 4.2 | Functional description | 24 |
| 4.2.1 | Structure | 24 |
| 4.2.2 | Modules structure | 25 |
| 4.2.3 | Low power mode | 27 |
| 4.3 | Write operation | 27 |
| 4.3.1 | Power supply drop | 28 |
| 4.4 | Registers description | 28 |
| 4.4.1 | Flash control register 0 low | 28 |
| 4.4.2 | Flash control register 0 high | 29 |
| 4.4.3 | Flash control register 1 low | 31 |
| 4.4.4 | Flash control register 1 high | 32 |
| 4.4.5 | Flash data register 0 low | 33 |
| 4.4.6 | Flash data register 0 high | 33 |
| 4.4.7 | Flash data register 1 low | 33 |
| 4.4.8 | Flash data register 1 high | 34 |
| 4.4.9 | Flash address register low | 34 |
| 4.4.10 | Flash address register high | 35 |
| 4.4.11 | Flash error register | 35 |
| 4.4.12 | XFlash interface control register | 36 |
| 4.5 | Protection strategy | 37 |
| 4.5.1 | Protection registers | 37 |
| 4.5.2 | Flash non volatile write protection X register low | 37 |
| 4.5.3 | Flash non volatile write protection X register high | 38 |
| 4.5.4 | Flash non volatile write protection I register low | 38 |
| 4.5.5 | Flash non volatile write protection I register high | 38 |
| 4.5.6 | Flash non volatile access protection register 0 | 39 |

| | | |
|----------|--|-----------|
| 4.5.7 | Flash non volatile access protection register 1 low | 39 |
| 4.5.8 | Flash non volatile access protection register 1 high | 40 |
| 4.5.9 | Access protection | 40 |
| 4.5.10 | Write protection | 41 |
| 4.5.11 | Temporary unprotection | 41 |
| 4.6 | Write operation examples | 42 |
| 4.7 | Write operation summary | 44 |
| 5 | Bootstrap loader | 46 |
| 5.1 | Selection among user-code, standard or alternate bootstrap | 46 |
| 5.2 | Standard bootstrap loader | 47 |
| 5.2.1 | Entering the standard bootstrap loader | 47 |
| 5.2.2 | ST10 configuration in BSL | 48 |
| 5.2.3 | Bootling steps | 50 |
| 5.2.4 | Hardware to activate BSL | 50 |
| 5.2.5 | Memory configuration in bootstrap loader mode | 51 |
| 5.2.6 | Loading the start-up code | 52 |
| 5.2.7 | Exiting bootstrap loader mode | 53 |
| 5.2.8 | Hardware requirements | 53 |
| 5.3 | Standard bootstrap with UART (RS232 or K-Line) | 53 |
| 5.3.1 | Features | 53 |
| 5.3.2 | Entering bootstrap via UART | 54 |
| 5.3.3 | ST10 Configuration in UART BSL (RS232 or K-Line) | 55 |
| 5.3.4 | Loading the start-up code | 55 |
| 5.3.5 | Choosing the baud rate for the BSL via UART | 56 |
| 5.4 | Standard bootstrap with CAN | 57 |
| 5.4.1 | Features | 57 |
| 5.4.2 | Entering the CAN bootstrap loader | 58 |
| 5.4.3 | ST10 configuration in CAN BSL | 59 |
| 5.4.4 | Loading the start-up code via CAN | 59 |
| 5.4.5 | Choosing the baud rate for the BSL via CAN | 60 |
| 5.4.6 | Computing the baud rate error | 63 |
| 5.4.7 | Bootstrap via CAN | 63 |
| 5.5 | Comparing the old and the new bootstrap loader | 64 |
| 5.5.1 | Software aspects | 64 |
| 5.5.2 | Hardware aspects | 65 |

| | | |
|-----------|---|-----------|
| 5.6 | Alternate boot mode (ABM) | 65 |
| 5.6.1 | Activation | 65 |
| 5.6.2 | Memory mapping | 65 |
| 5.6.3 | Interrupts | 65 |
| 5.6.4 | ST10 configuration in alternate boot mode | 66 |
| 5.6.5 | Watchdog | 66 |
| 5.6.6 | Exiting alternate boot mode | 66 |
| 5.6.7 | Alternate boot user software | 67 |
| 5.6.8 | User/alternate mode signature integrity check | 67 |
| 5.6.9 | Alternate boot user software aspects | 67 |
| 5.6.10 | EMUCON register | 67 |
| 5.6.11 | Internal decoding of test modes | 68 |
| 5.6.12 | Example | 68 |
| 5.7 | Selective boot mode | 68 |
| 6 | Central processing unit (CPU) | 71 |
| 6.1 | Multiplier-accumulator unit (MAC) | 72 |
| 6.2 | Instruction set summary | 73 |
| 6.3 | MAC coprocessor specific instructions | 75 |
| 7 | External bus controller | 76 |
| 8 | Interrupt system | 77 |
| 8.1 | X-Peripheral interrupt | 79 |
| 8.2 | Exception and error traps list | 81 |
| 9 | Capture / compare (CAPCOM) units | 82 |
| 10 | General purpose timer unit | 84 |
| 10.1 | GPT1 | 84 |
| 10.2 | GPT2 | 86 |
| 11 | PWM modules | 88 |
| 12 | Parallel ports | 89 |
| 12.1 | Introduction | 89 |
| 12.2 | I/O's special features | 90 |

| | | | |
|-----------|--------|--|------------|
| | 12.2.1 | Open drain mode | 90 |
| | 12.2.2 | Input threshold control | 90 |
| | 12.3 | Alternate port functions | 90 |
| 13 | | A/D converter | 92 |
| 14 | | Serial channels | 94 |
| | 14.1 | Asynchronous / synchronous serial interfaces | 94 |
| | 14.2 | ASCx in asynchronous mode | 94 |
| | 14.3 | ASCx in synchronous mode | 95 |
| | 14.4 | High speed synchronous serial interfaces | 96 |
| 15 | | I2C interface | 98 |
| 16 | | CAN modules | 99 |
| | 16.1 | Configuration support | 99 |
| | 16.2 | CAN bus configurations | 100 |
| 17 | | Real time clock | 102 |
| 18 | | Watchdog timer | 103 |
| 19 | | System reset | 104 |
| | 19.1 | Input filter | 104 |
| | 19.2 | Asynchronous reset | 105 |
| | 19.3 | Synchronous reset (warm reset) | 110 |
| | 19.4 | Software reset | 116 |
| | 19.5 | Watchdog timer reset | 117 |
| | 19.6 | Bidirectional reset | 118 |
| | 19.7 | Reset circuitry | 122 |
| | 19.8 | Reset application examples | 125 |
| | 19.9 | Reset summary | 127 |
| 20 | | Power reduction modes | 130 |
| | 20.1 | Idle mode | 130 |
| | 20.2 | Power down mode | 130 |

| | | |
|-----------|---|------------|
| 20.2.1 | Protected power down mode | 131 |
| 20.2.2 | Interruptible power down mode | 131 |
| 20.3 | Stand-by mode | 131 |
| 20.3.1 | Entering stand-by mode | 132 |
| 20.3.2 | Exiting stand-by mode | 133 |
| 20.3.3 | Real time clock and stand-by mode | 133 |
| 20.3.4 | Power reduction modes summary | 134 |
| 21 | Programmable output clock divider | 135 |
| 22 | Register set | 136 |
| 22.1 | Register description format | 136 |
| 22.2 | General purpose registers (GPRs) | 137 |
| 22.3 | Special function registers ordered by name | 139 |
| 22.4 | Special function registers ordered by address | 146 |
| 22.5 | X-registers sorted by name | 153 |
| 22.6 | X-registers ordered by address | 158 |
| 22.7 | Flash registers ordered by name | 163 |
| 22.8 | Flash registers ordered by address | 164 |
| 22.9 | Identification registers | 165 |
| 22.10 | System configuration registers | 167 |
| 22.10.1 | XPERCON and XPEREMU registers | 174 |
| 22.11 | Emulation dedicated registers | 175 |
| 23 | Electrical characteristics | 176 |
| 23.1 | Absolute maximum ratings | 176 |
| 23.2 | Recommended operating conditions | 176 |
| 23.3 | Power considerations | 177 |
| 23.4 | Parameter interpretation | 178 |
| 23.5 | DC characteristics | 178 |
| 23.6 | Flash characteristics | 183 |
| 23.7 | A/D converter characteristics | 185 |
| 23.7.1 | Conversion timing control | 186 |
| 23.7.2 | A/D conversion accuracy | 187 |
| 23.7.3 | Total unadjusted error | 188 |

| | | |
|---------|--|------------|
| 23.7.4 | Analog reference pins | 189 |
| 23.7.5 | Analog input pins | 189 |
| 23.7.6 | Example of external network sizing | 193 |
| 23.8 | AC characteristics | 194 |
| 23.8.1 | Test waveforms | 194 |
| 23.8.2 | Definition of internal timing | 195 |
| 23.8.3 | Clock generation modes | 196 |
| 23.8.4 | Prescaler operation | 196 |
| 23.8.5 | Direct drive | 196 |
| 23.8.6 | Oscillator watchdog (OWD) | 197 |
| 23.8.7 | Phase locked loop (PLL) | 197 |
| 23.8.8 | Voltage controlled oscillator | 198 |
| 23.8.9 | PLL Jitter | 199 |
| 23.8.10 | Jitter in the input clock | 199 |
| 23.8.11 | Noise in the PLL loop | 199 |
| 23.8.12 | PLL lock/unlock | 201 |
| 23.8.13 | Main oscillator specifications | 201 |
| 23.8.14 | 32 kHz Oscillator specifications | 202 |
| 23.8.15 | External clock drive XTAL1 | 203 |
| 23.8.16 | Memory cycle variables | 204 |
| 23.8.17 | External memory bus timing | 205 |
| 23.8.18 | Multiplexed bus | 206 |
| 23.8.19 | Demultiplexed bus | 212 |
| 23.8.20 | CLKOUT and READY | 218 |
| 23.8.21 | External bus arbitration | 220 |
| 23.8.22 | High-speed synchronous serial interface (SSC) timing modes | 222 |
| 24 | Package information | 226 |
| 25 | Revision history | 228 |

List of tables

| | | |
|-----------|---|----|
| Table 1. | Pin description | 17 |
| Table 2. | Flash modules absolute mapping | 24 |
| Table 3. | Flash modules sectorization (read operations) | 25 |
| Table 4. | Flash modules sectorization (write operations or with roms1='1') | 26 |
| Table 5. | Control register interface | 27 |
| Table 6. | Flash control register 0 low | 28 |
| Table 7. | Flash control register 0 high | 29 |
| Table 8. | Flash control register 1 low | 31 |
| Table 9. | Flash control register 1 high | 32 |
| Table 10. | Banks (BxS) and sectors (BxFy) status bits meaning | 33 |
| Table 11. | Flash data register 0 low | 33 |
| Table 12. | Flash data register 0 high | 33 |
| Table 13. | Flash data register 1 low | 34 |
| Table 14. | Flash data register 1 high | 34 |
| Table 15. | Flash address register low | 34 |
| Table 16. | Flash address register high | 35 |
| Table 17. | Flash error register | 35 |
| Table 18. | XFlash interface control register | 36 |
| Table 19. | Flash non volatile write protection X register low | 37 |
| Table 20. | Flash non volatile write protection X register high | 38 |
| Table 21. | Flash non volatile write protection I register low | 38 |
| Table 22. | Flash non volatile write protection I register high | 38 |
| Table 23. | Flash non volatile access protection register 0 | 39 |
| Table 24. | Flash non volatile access protection register 1 low | 39 |
| Table 25. | Flash non volatile access protection register 1 high | 40 |
| Table 26. | Summary of access protection level | 40 |
| Table 27. | Flash write operations | 44 |
| Table 28. | ST10F276 boot mode selection | 47 |
| Table 29. | ST10 configuration in BSL mode | 48 |
| Table 30. | ST10 configuration in UART BSL mode (RS232 or K-line) | 55 |
| Table 31. | ST10 configuration in CAN BSL | 59 |
| Table 32. | BRP and PT0 values | 62 |
| Table 33. | Software topics summary | 64 |
| Table 34. | Hardware topics summary | 65 |
| Table 35. | ST10 configuration in alternate boot mode | 66 |
| Table 36. | ABM bit description | 68 |
| Table 37. | Selective boot | 69 |
| Table 38. | Standard instruction set summary | 73 |
| Table 39. | MAC instruction set summary | 75 |
| Table 40. | Interrupt sources | 77 |
| Table 41. | X-Interrupt detailed mapping | 80 |
| Table 42. | Trap priorities | 81 |
| Table 43. | Compare modes | 83 |
| Table 44. | CAPCOM timer input frequencies, resolutions and periods at 40 MHz | 83 |
| Table 45. | CAPCOM timer input frequencies, resolutions and periods at 64 MHz | 83 |
| Table 46. | GPT1 timer input frequencies, resolutions and periods at 40 MHz | 84 |
| Table 47. | GPT1 timer input frequencies, resolutions and periods at 64 MHz | 85 |
| Table 48. | GPT2 timer input frequencies, resolutions and periods at 40 MHz | 86 |

| | | |
|------------|--|-----|
| Table 49. | GPT2 timer input frequencies, resolutions and periods at 64 MHz. | 86 |
| Table 50. | PWM unit frequencies and resolutions at 40 MHz CPU clock | 88 |
| Table 51. | PWM unit frequencies and resolutions at 64 MHz CPU clock | 88 |
| Table 52. | ASC asynchronous baud rates by reload value and deviation errors (fCPU = 40 MHz) . . | 94 |
| Table 53. | ASC asynchronous baud rates by reload value and deviation errors (fCPU = 64 MHz) . . | 95 |
| Table 54. | ASC synchronous baud rates by reload value and deviation errors (fCPU = 40 MHz) . . | 95 |
| Table 55. | ASC synchronous baud rates by reload value and deviation errors (fCPU = 64 MHz) . . | 96 |
| Table 56. | Synchronous baud rate and reload values (fCPU = 40 MHz). | 97 |
| Table 57. | Synchronous baud rate and reload values (fCPU = 64 MHz). | 97 |
| Table 58. | WDTRREL reload value (fCPU = 40 MHz) | 103 |
| Table 59. | WDTRREL reload value (fCPU = 64 MHz) | 103 |
| Table 60. | Reset event definition | 104 |
| Table 61. | Reset event. | 127 |
| Table 62. | PORT0 latched configuration for the different reset events | 128 |
| Table 63. | Power reduction modes summary | 134 |
| Table 64. | Description. | 136 |
| Table 65. | General purpose registers (GPRs) | 137 |
| Table 66. | General purpose registers (GPRs) bytewise addressing. | 137 |
| Table 67. | Special function registers ordered by address. | 139 |
| Table 68. | Special function registers ordered by address. | 146 |
| Table 69. | X-Registers ordered by name. | 153 |
| Table 70. | X-registers ordered by address | 158 |
| Table 71. | Flash registers ordered by name | 163 |
| Table 72. | FLASH registers ordered by address | 164 |
| Table 73. | MANUF description. | 165 |
| Table 74. | IDCHIP description | 165 |
| Table 75. | IDMEM description | 166 |
| Table 76. | IDPROG description | 166 |
| Table 77. | SYSCON description | 167 |
| Table 78. | BUSCON4 description | 169 |
| Table 79. | RPOH description | 170 |
| Table 80. | EXIXES bit description | 171 |
| Table 81. | EXISEL | 171 |
| Table 82. | EXIXSS and port 2 pin configurations | 171 |
| Table 83. | SFR area description | 172 |
| Table 84. | ESFR description | 172 |
| Table 85. | Segment 8 address range mapping | 174 |
| Table 86. | Absolute maximum ratings | 176 |
| Table 87. | Recommended operating conditions | 176 |
| Table 88. | Thermal characteristics. | 177 |
| Table 89. | Package characteristics | 178 |
| Table 90. | DC characteristics. | 178 |
| Table 91. | Flash characteristics | 183 |
| Table 92. | Data retention characteristics | 184 |
| Table 94. | A/D Converter programming. | 186 |
| Table 95. | On-chip clock generator selections. | 196 |
| Table 96. | Internal PLL divider mechanism | 198 |
| Table 97. | PLL lock/unlock timing | 201 |
| Table 98. | Main oscillator specifications | 201 |
| Table 99. | Negative resistance (absolute min. value @ 125oC / VDD = 4.5V). | 202 |
| Table 100. | 32 kHz Oscillator specifications | 202 |
| Table 101. | Minimum values of negative resistance (module). | 203 |

| | | |
|------------|---------------------------------------|-----|
| Table 102. | External clock drive timing | 204 |
| Table 103. | Memory cycle variables | 204 |
| Table 104. | Multiplexed bus | 206 |
| Table 105. | Demultiplexed bus | 212 |
| Table 106. | CLKOUT and READY | 218 |
| Table 107. | External bus arbitration | 220 |
| Table 108. | Master mode | 222 |
| Table 109. | Slave mode | 224 |
| Table 110. | Document revision history | 228 |

List of figures

| | | |
|------------|--|-----|
| Figure 1. | Logic symbol | 15 |
| Figure 2. | Pin configuration (top view) | 16 |
| Figure 3. | Block diagram | 23 |
| Figure 4. | Flash modules structure | 24 |
| Figure 5. | ST10F276 new standard bootstrap loader program flow | 49 |
| Figure 6. | Bootling steps for ST10F276 | 50 |
| Figure 7. | Hardware provisions to activate the BSL | 51 |
| Figure 8. | Memory configuration after reset | 52 |
| Figure 9. | UART bootstrap loader sequence | 54 |
| Figure 10. | Baud rate deviation between host and ST10F276 | 56 |
| Figure 11. | CAN bootstrap loader sequence | 57 |
| Figure 12. | Bit rate measurement over a predefined zero-frame | 60 |
| Figure 13. | Reset boot sequence | 70 |
| Figure 14. | CPU Block Diagram (MAC Unit not included). | 71 |
| Figure 15. | MAC unit architecture | 72 |
| Figure 16. | X-Interrupt basic structure | 80 |
| Figure 17. | Block diagram of GPT1 | 85 |
| Figure 18. | Block diagram of GPT2 | 87 |
| Figure 19. | Block diagram of PWM module | 88 |
| Figure 20. | Connection to single CAN bus via separate CAN transceivers | 100 |
| Figure 21. | Connection to single CAN bus via common CAN transceivers | 100 |
| Figure 22. | Connection to two different CAN buses (e.g. for gateway application) | 101 |
| Figure 23. | Connection to one CAN bus with internal Parallel Mode enabled | 101 |
| Figure 24. | Asynchronous power-on RESET (EA = 1) | 107 |
| Figure 25. | Asynchronous power-on RESET (EA = 0) | 108 |
| Figure 26. | Asynchronous hardware RESET (EA = 1) | 109 |
| Figure 27. | Asynchronous hardware RESET (EA = 0) | 110 |
| Figure 28. | Synchronous short / long hardware RESET (EA = 1) | 113 |
| Figure 29. | Synchronous short / long hardware RESET (EA = 0) | 114 |
| Figure 30. | Synchronous long hardware RESET (EA = 1) | 115 |
| Figure 31. | Synchronous long hardware RESET (EA = 0) | 116 |
| Figure 32. | SW / WDT unidirectional RESET (EA = 1) | 117 |
| Figure 33. | SW / WDT unidirectional RESET (EA = 0) | 118 |
| Figure 34. | SW / WDT bidirectional RESET (EA=1) | 120 |
| Figure 35. | SW / WDT bidirectional RESET (EA = 0) | 121 |
| Figure 36. | SW / WDT bidirectional RESET (EA=0) followed by a HW RESET | 122 |
| Figure 37. | Minimum external reset circuitry | 123 |
| Figure 38. | System reset circuit | 124 |
| Figure 39. | Internal (simplified) reset circuitry | 124 |
| Figure 40. | Example of software or watchdog bidirectional reset (EA = 1) | 125 |
| Figure 41. | Example of software or watchdog bidirectional reset (EA = 0) | 126 |
| Figure 42. | PORT0 bits latched into the different registers after reset | 129 |
| Figure 43. | External RC circuitry on RPD pin | 131 |
| Figure 44. | Port2 test mode structure | 182 |
| Figure 45. | Supply current versus the operating frequency (RUN and IDLE modes) | 182 |
| Figure 46. | A/D conversion characteristic | 188 |
| Figure 47. | A/D converter input pins scheme | 189 |
| Figure 48. | Charge sharing timing diagram during sampling phase | 190 |

| | | |
|------------|---|-----|
| Figure 49. | Anti-aliasing filter and conversion rate | 192 |
| Figure 50. | Input/output waveforms | 194 |
| Figure 51. | Float waveforms | 194 |
| Figure 52. | Generation mechanisms for the CPU clock | 195 |
| Figure 53. | ST10F276 PLL jitter | 200 |
| Figure 54. | Crystal oscillator and resonator connection diagram | 202 |
| Figure 55. | 32 kHz crystal oscillator connection diagram | 203 |
| Figure 56. | External clock drive XTAL1 | 204 |
| Figure 57. | Multiplexed bus with/without R/W delay and normal ALE | 208 |
| Figure 58. | Multiplexed bus with/without R/W delay and extended ALE | 209 |
| Figure 59. | Multiplexed bus, with/without R/W delay, normal ALE, R/W CS | 210 |
| Figure 60. | Multiplexed bus, with/without R/ W delay, extended ALE, R/W CS | 211 |
| Figure 61. | Demultiplexed bus, with/without read/write delay and normal ALE | 214 |
| Figure 62. | Demultiplexed bus with/without R/W delay and extended ALE | 215 |
| Figure 63. | Demultiplexed bus with ALE and R/W CS | 216 |
| Figure 64. | Demultiplexed bus, no R/W delay, extended ALE, R/W CS | 217 |
| Figure 65. | CLKOUT and READY | 219 |
| Figure 66. | External bus arbitration (releasing the bus) | 220 |
| Figure 67. | External bus arbitration (regaining the bus) | 221 |
| Figure 68. | SSC master timing | 223 |
| Figure 69. | SSC slave timing | 225 |
| Figure 70. | 144-pin plastic quad flat package | 226 |
| Figure 71. | 144-pin low profile quad flat package (10x10) | 227 |

1 Introduction

The ST10F276 is a derivative of the STMicroelectronics ST10 family of 16-bit single-chip CMOS microcontrollers. It combines high CPU performance (up to 32 million instructions per second) with high peripheral functionality and enhanced I/O-capabilities. It also provides on-chip high-speed single voltage Flash memory, on-chip high-speed RAM, and clock generation via PLL.

ST10F276 is processed in 0.18μm CMOS technology. The MCU core and the logic is supplied with a 5V to 1.8V on-chip voltage regulator. The part is supplied with a single 5V supply and I/Os work at 5V.

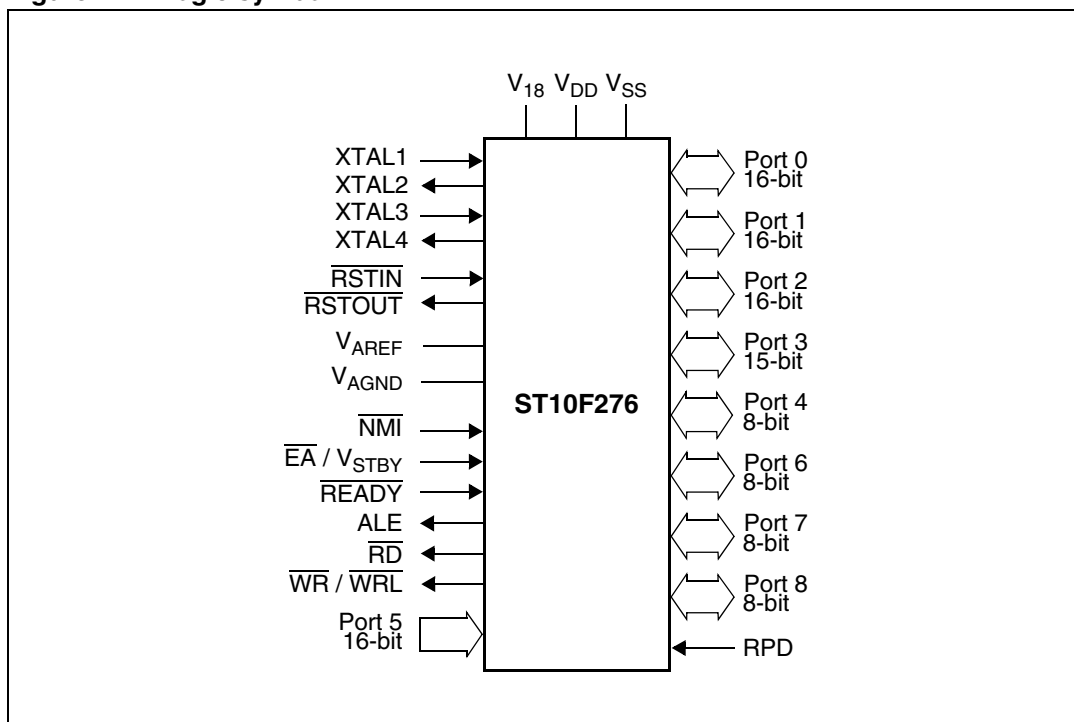
The device is upward compatible with the ST10F269 device, with the following set of differences:

- Flash control interface is now based on STMicroelectronics third generation of stand-alone Flash memories (M29F400 series), with an embedded Program/Erase Controller. This completely frees up the CPU during programming or erasing the Flash.
- Only one supply pin (ex DC1 in ST10F269, renamed into V_{18}) on the QFP144 package is used for decoupling the internally generated 1.8V core logic supply. Do not connect this pin to 5.0V external supply. Instead, this pin should be connected to a decoupling capacitor (ceramic type, typical value 10nF, maximum value 100nF).
- The AC and DC parameters are modified due to a difference in the maximum CPU frequency.
- A new V_{DD} pin replaces DC2 of ST10F269.
- \overline{EA} pin assumes a new alternate functionality: it is also used to provide a dedicated power supply (see V_{STBY}) to maintain biased a portion of the XRAM (16Kbytes) when the main Power Supply of the device (V_{DD} and consequently the internally generated V_{18}) is turned off for low power mode, allowing data retention. V_{STBY} voltage shall be in the range 4.5-5.5 Volt, and a dedicated embedded low power voltage regulator is in charge to provide the 1.8V for the RAM, the low-voltage section of the 32kHz oscillator and the Real Time Clock module when not disabled. It is allowed to exceed the upper limit up to 6V for a very short period of time during the global life of the device, and exceed the lower limit down to 4V when RTC and 32kHz on-chip oscillator are not used.
- A second SSC mapped on the XBUS is added (SSC of ST10F269 becomes here SSC0, while the new one is referred as XSSC or simply SSC1). Note that some restrictions and functional differences due to the XBUS peculiarities are present between the classic SSC, and the new XSSC.
- A second ASC mapped on the XBUS is added (ASC0 of ST10F269 remains ASC0, while the new one is referred as XASC or simply as ASC1). Note that some restrictions and functional differences due to the XBUS peculiarities are present between the classic ASC, and the new XASC.
- A second PWM mapped on the XBUS is added (PWM of ST10F269 becomes here PWM0, while the new one is referred as XPWM or simply as PWM1). Note that some

restrictions and functional differences due to the XBUS peculiarities are present between the classic PWM, and the new XPWM.

- An I²C interface on the XBUS is added (see X-I²C or simply I²C interface).
- CLKOUT function can output either the CPU clock (like in ST10F269) or a software programmable prescaled value of the CPU clock.
- Embedded memory size has been significantly increased (both Flash and RAM).
- PLL multiplication factors have been adapted to new frequency range.
- A/D Converter is not fully compatible versus ST10F269 (timing and programming model). Formula for the conversion time is still valid, while the sampling phase programming model is different.
Besides, additional 8 channels are available on P1L pins as alternate function: the accuracy reachable with these extra channels is reduced with respect to the standard Port5 channels.
- External Memory bus potential limitations on maximum speed and maximum capacitance load could be introduced (under evaluation): ST10F276 will probably not be able to address an external memory at 64MHz with 0 wait states (under evaluation).
- XPERCON register bit mapping modified according to new peripherals implementation (not fully compatible with ST10F269).
- Bondout chip for emulation (ST10R201) cannot achieve more than 50MHz at room temperature (so no real time emulation possible at maximum speed).
- Input section characteristics are different. The threshold programmability is extended to all port pins (additional XPICON register); it is possible to select standard TTL (with up to 500mV of hysteresis) and standard CMOS (with up to 800mV of hysteresis).
- Output transition is not programmable.
- CAN module is enhanced: ST10F276 implements two C-CAN modules, so the programming model is slightly different. Besides, the possibility to map in parallel the two CAN modules is added (on P4.5/P4.6).
- On-chip main oscillator input frequency range has been reshaped, reducing it from 1-25MHz down to 4-12MHz. This is a high performance oscillator amplifier, providing a very high negative resistance and wide oscillation amplitude: when this on-chip amplifier is used as reference for Real Time Clock module, the Power-down consumption is dominated by the consumption of the oscillator amplifier itself. A metal option is added to offer a low power oscillator amplifier working in the range of 4-8MHz: this will allow a power consumption reduction when Real Time Clock is running in Power Down mode using as reference the on-chip main oscillator clock.
- A second on-chip oscillator amplifier circuit (32kHz) is implemented for low power modes: it can be used to provide the reference to the Real Time Clock counter (either in Power Down or Stand-by mode). Pin XTAL3 and XTAL4 replace a couple of V_{DD}/V_{SS} pins of ST10F269.
- Possibility to re-program internal XBUS chip select window characteristics (XRAM2 and XFLASH address window) is added.

Figure 1. Logic symbol



2 Pin data

Figure 2. Pin configuration (top view)

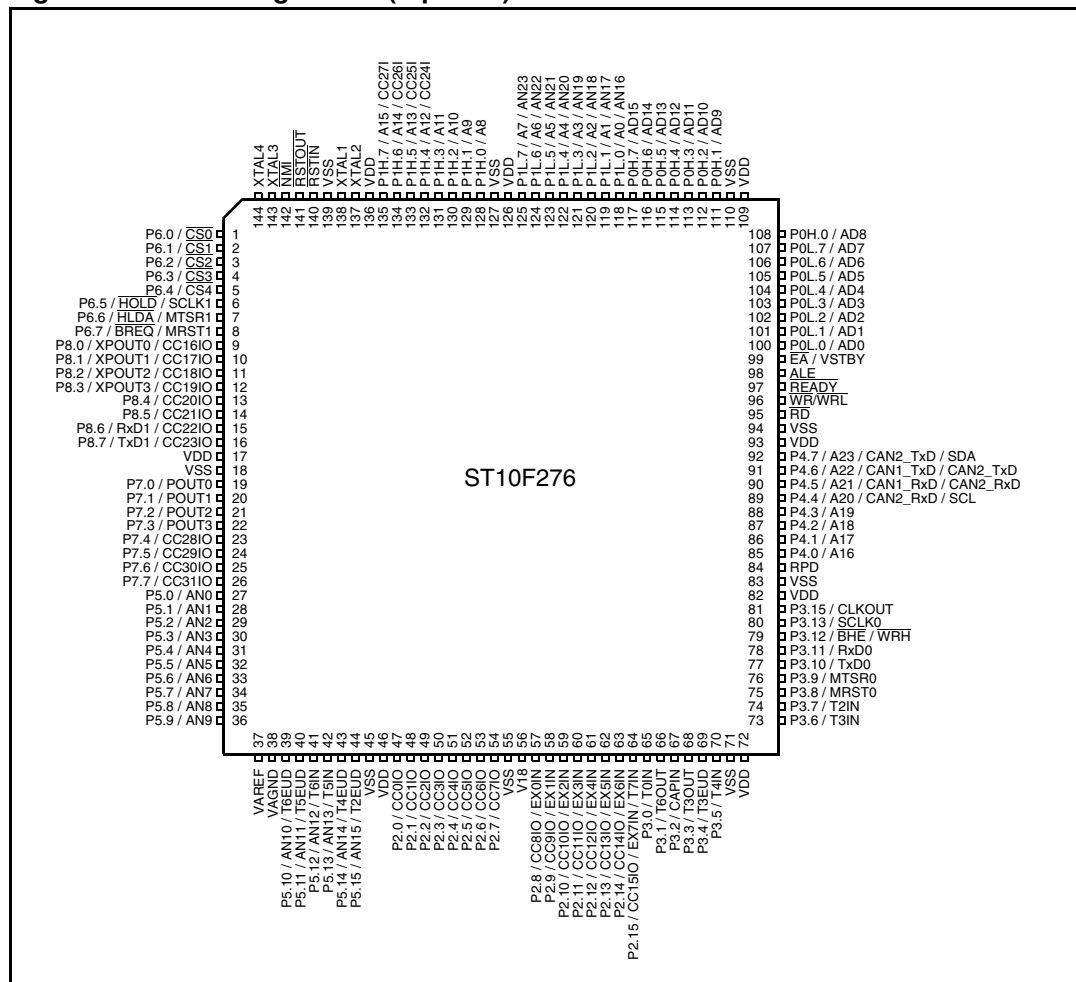


Table 1. Pin description

| Symbol | Pin | Type | Function | | |
|-------------|-------|------|--|-------------------|--|
| P6.0 - P6.7 | 1 - 8 | I/O | 8-bit bidirectional I/O port, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. Port 6 outputs can be configured as push-pull or open drain drivers. The input threshold of Port 6 is selectable (TTL or CMOS). The following Port 6 pins have alternate functions: | | |
| | 1 | O | P6.0 | $\overline{CS0}$ | Chip select 0 output |
| | ... | ... | ... | ... | ... |
| | 5 | O | P6.4 | $\overline{CS4}$ | Chip select 4 output |
| | 6 | I | P6.5 | \overline{HOLD} | External master hold request input |
| | | I/O | | SCLK1 | SSC1: master clock output / slave clock input |
| | 7 | O | P6.6 | \overline{HLDA} | Hold acknowledge output |
| | | I/O | | MTSR1 | SSC1: master-transmitter / slave-receiver O/I |
| | 8 | O | P6.7 | BREQ | Bus request output |
| | | I/O | | MRST1 | SSC1: master-receiver / slave-transmitter I/O |
| P8.0 - P8.7 | 9-16 | I/O | 8-bit bidirectional I/O port, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. Port 8 outputs can be configured as push-pull or open drain drivers. The input threshold of Port 8 is selectable (TTL or CMOS). The following Port 8 pins have alternate functions: | | |
| | 9 | I/O | P8.0 | CC16IO | CAPCOM2: CC16 capture input / compare output |
| | | O | | XPWM0 | PWM1: channel 0 output |
| | ... | ... | ... | ... | ... |
| | 12 | I/O | P8.3 | CC19IO | CAPCOM2: CC19 capture input / compare output |
| | | O | | XPWM0 | PWM1: channel 3 output |
| | 13 | I/O | P8.4 | CC20IO | CAPCOM2: CC20 capture input / compare output |
| | 14 | I/O | P8.5 | CC21IO | CAPCOM2: CC21 capture input / compare output |
| | 15 | I/O | P8.6 | CC22IO | CAPCOM2: CC22 capture input / compare output |
| | | I/O | | RxD1 | ASC1: Data input (Asynchronous) or I/O (Synchronous) |
| | 16 | I/O | P8.7 | CC23IO | CAPCOM2: CC23 capture input / compare output |
| | | O | | TxD1 | ASC1: Clock / Data output (Asynchronous/Synchronous) |

Table 1. Pin description (continued)

| Symbol | Pin | Type | Function | | |
|------------------------------|----------------|--------|---|--------|---|
| P7.0 - P7.7 | 19-26 | I/O | 8-bit bidirectional I/O port, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. Port 7 outputs can be configured as push-pull or open drain drivers. The input threshold of Port 7 is selectable (TTL or CMOS). The following Port 7 pins have alternate functions: | | |
| | 19 | O | P7.0 | POUT0 | PWM0: channel 0 output |
| | ... | ... | ... | ... | ... |
| | 22 | O | P7.3 | POUT3 | PWM0: channel 3 output |
| | 23 | I/O | P7.4 | CC28IO | CAPCOM2: CC28 capture input / compare output |
| | ... | ... | ... | ... | ... |
| | 26 | I/O | P7.7 | CC31IO | CAPCOM2: CC31 capture input / compare output |
| P5.0 - P5.9 P5.10 - P5.15 | 27-36 39-44 | I I | 16-bit input-only port with Schmitt-Trigger characteristics. The pins of Port 5 can be the analog input channels (up to 16) for the A/D converter, where P5.x equals ANx (Analog input channel x), or they are timer inputs. The input threshold of Port 5 is selectable (TTL or CMOS). The following Port 5 pins have alternate functions: | | |
| | 39 | I | P5.10 | T6EUD | GPT2: timer T6 external up/down control input |
| | 40 | I | P5.11 | T5EUD | GPT2: timer T5 external up/down control input |
| | 41 | I | P5.12 | T6IN | GPT2: timer T6 count input |
| | 42 | I | P5.13 | T5IN | GPT2: timer T5 count input |
| | 43 | I | P5.14 | T4EUD | GPT1: timer T4 external up/down control input |
| | 44 | I | P5.15 | T2EUD | GPT1: timer T2 external up/down control input |
| P2.0 - P2.7 P2.8 - P2.15 | 47-54 57-64 | I/O | 16-bit bidirectional I/O port, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. Port 2 outputs can be configured as push-pull or open drain drivers. The input threshold of Port 2 is selectable (TTL or CMOS). The following Port 2 pins have alternate functions: | | |
| | 47 | I/O | P2.0 | CC0IO | CAPCOM: CC0 capture input/compare output |
| | ... | ... | ... | ... | ... |
| | 54 | I/O | P2.7 | CC7IO | CAPCOM: CC7 capture input/compare output |
| | 57 | I/O | P2.8 | CC8IO | CAPCOM: CC8 capture input/compare output |
| | | I | | EX0IN | Fast external interrupt 0 input |
| | ... | ... | ... | ... | ... |
| | 64 | I/O | P2.15 | CC15IO | CAPCOM: CC15 capture input/compare output |
| | | I | | EX7IN | Fast external interrupt 7 input |
| | | I | | T7IN | CAPCOM2: timer T7 count input |

Table 1. Pin description (continued)

| Symbol | Pin | Type | Function | | |
|---------------------------------------|------------------------|-------------------|--|-------------------------|---|
| P3.0 - P3.5 P3.6 - P3.13, P3.15 | 65-70, 73-80, 81 | I/O I/O I/O | 15-bit (P3.14 is missing) bidirectional I/O port, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. Port 3 outputs can be configured as push-pull or open drain drivers. The input threshold of Port 3 is selectable (TTL or CMOS). The following Port 3 pins have alternate functions: | | |
| | 65 | I | P3.0 | T0IN | CAPCOM1: timer T0 count input |
| | 66 | O | P3.1 | T6OUT | GPT2: timer T6 toggle latch output |
| | 67 | I | P3.2 | CAPIN | GPT2: register CAPREL capture input |
| | 68 | O | P3.3 | T3OUT | GPT1: timer T3 toggle latch output |
| | 69 | I | P3.4 | T3EUD | GPT1: timer T3 external up/down control input |
| | 70 | I | P3.5 | T4IN | GPT1; timer T4 input for count/gate/reload/capture |
| | 73 | I | P3.6 | T3IN | GPT1: timer T3 count/gate input |
| | 74 | I | P3.7 | T2IN | GPT1: timer T2 input for count/gate/reload / capture |
| | 75 | I/O | P3.8 | MRST0 | SSC0: master-receiver/slave-transmitter I/O |
| | 76 | I/O | P3.9 | MTSR0 | SSC0: master-transmitter/slave-receiver O/I |
| | 77 | O | P3.10 | TxD0 | ASC0: clock / data output (asynchronous/synchronous) |
| | 78 | I/O | P3.11 | RxD0 | ASC0: data input (asynchronous) or I/O (synchronous) |
| | 79 | O | P3.12 | $\overline{\text{BHE}}$ | External memory high byte enable signal |
| | | | | $\overline{\text{WRH}}$ | External memory high byte write strobe |
| | 80 | I/O | P3.13 | SCLK0 | SSC0: master clock output / slave clock input |
| | 81 | O | P3.15 | CLKOUT | System clock output (programmable divider on CPU clock) |

Table 1. Pin description (continued)

| Symbol | Pin | Type | Function | | |
|--|-------|------|--|----------|--|
| P4.0 –P4.7 | 85-92 | I/O | Port 4 is an 8-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. The input threshold is selectable (TTL or CMOS). Port 4.4, 4.5, 4.6 and 4.7 outputs can be configured as push-pull or open drain drivers. In case of an external bus configuration, Port 4 can be used to output the segment address lines: | | |
| | 85 | O | P4.0 | A16 | Segment address line |
| | 86 | O | P4.1 | A17 | Segment address line |
| | 87 | O | P4.2 | A18 | Segment address line |
| | 88 | O | P4.3 | A19 | Segment address line |
| | 89 | O | P4.4 | A20 | Segment address line |
| | | I | | CAN2_RxD | CAN2: receive data input |
| | | I/O | | SCL | I ² C Interface: serial clock |
| | 90 | O | P4.5 | A21 | Segment address line |
| | | I | | CAN1_RxD | CAN1: receive data input |
| | | I | | CAN2_RxD | CAN2: receive data input |
| | 91 | O | P4.6 | A22 | Segment address line |
| | | O | | CAN1_TxD | CAN1: transmit data output |
| | | O | | CAN2_TxD | CAN2: transmit data output |
| | 92 | O | P4.7 | A23 | Most significant segment address line |
| | | O | | CAN2_TxD | CAN2: transmit data output |
| | | I/O | | SDA | I ² C Interface: serial data |
| $\overline{\text{RD}}$ | 95 | O | External memory read strobe. $\overline{\text{RD}}$ is activated for every external instruction or data read access. | | |
| $\overline{\text{WR}}/\text{WRL}$ | 96 | O | External memory write strobe. In $\overline{\text{WR}}$ -mode this pin is activated for every external data write access. In WRL mode this pin is activated for low byte data write accesses on a 16-bit bus, and for every data write access on an 8-bit bus. See WRCFG in the SYSCON register for mode selection. | | |
| $\text{READY}/\overline{\text{READY}}$ | 97 | I | Ready input. The active level is programmable. When the ready function is enabled, the selected inactive level at this pin, during an external memory access, will force the insertion of waitstate cycles until the pin returns to the selected active level. | | |
| ALE | 98 | O | Address latch enable output. In case of use of external addressing or of multiplexed mode, this signal is the latch command of the address lines. | | |

Table 1. Pin description (continued)

| Symbol | Pin | Type | Function | | | | | | | | | | | | | | | | | | | | |
|--|-----------------------------|------------|--|--------|-----------------------------|-----------------|-------|--------|----------------|---------|---------|----------------|-----|----------|-----------------|-------|--------|----------------|-----------|-----------|----------------|----------|------------|
| \overline{EA} / V_{STBY} | 99 | I | <p>External access enable pin.</p> <p>A low level applied to this pin during and after Reset forces the ST10F276 to start the program from the external memory space. A high level forces ST10F276 to start in the internal memory space. This pin is also used (when Stand-by mode is entered, that is ST10F276 under reset and main V_{DD} turned off) to bias the 32 kHz oscillator amplifier circuit and to provide a reference voltage for the low-power embedded voltage regulator which generates the internal 1.8V supply for the RTC module (when not disabled) and to retain data inside the Stand-by portion of the XRAM (16Kbyte).</p> <p>It can range from 4.5 to 5.5V (6V for a reduced amount of time during the device life, 4.0V when RTC and 32 kHz on-chip oscillator amplifier are turned off). In running mode, this pin can be tied low during reset without affecting 32 kHz oscillator, RTC and XRAM activities, since the presence of a stable V_{DD} guarantees the proper biasing of all those modules.</p> | | | | | | | | | | | | | | | | | | | | |
| P0L.0 - P0L.7, P0H.0 P0H.1 - P0H.7 | 100-107, 108, 111-117 | I/O | <p>Two 8-bit bidirectional I/O ports P0L and P0H, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. The input threshold of Port 0 is selectable (TTL or CMOS).</p> <p>In case of an external bus configuration, PORT0 serves as the address (A) and as the address / data (AD) bus in multiplexed bus modes and as the data (D) bus in demultiplexed bus modes.</p> <p>Demultiplexed bus modes</p> <table><tr><td>Data path width</td><td>8-bit</td><td>16-bit</td></tr><tr><td>P0L.0 – P0L.7:</td><td>D0 – D7</td><td>D0 - D7</td></tr><tr><td>P0H.0 – P0H.7:</td><td>I/O</td><td>D8 - D15</td></tr></table> <p>Multiplexed bus modes</p> <table><tr><td>Data path width</td><td>8-bit</td><td>16-bit</td></tr><tr><td>P0L.0 – P0L.7:</td><td>AD0 – AD7</td><td>AD0 - AD7</td></tr><tr><td>P0H.0 – P0H.7:</td><td>A8 – A15</td><td>AD8 - AD15</td></tr></table> | | | Data path width | 8-bit | 16-bit | P0L.0 – P0L.7: | D0 – D7 | D0 - D7 | P0H.0 – P0H.7: | I/O | D8 - D15 | Data path width | 8-bit | 16-bit | P0L.0 – P0L.7: | AD0 – AD7 | AD0 - AD7 | P0H.0 – P0H.7: | A8 – A15 | AD8 - AD15 |
| Data path width | 8-bit | 16-bit | | | | | | | | | | | | | | | | | | | | | |
| P0L.0 – P0L.7: | D0 – D7 | D0 - D7 | | | | | | | | | | | | | | | | | | | | | |
| P0H.0 – P0H.7: | I/O | D8 - D15 | | | | | | | | | | | | | | | | | | | | | |
| Data path width | 8-bit | 16-bit | | | | | | | | | | | | | | | | | | | | | |
| P0L.0 – P0L.7: | AD0 – AD7 | AD0 - AD7 | | | | | | | | | | | | | | | | | | | | | |
| P0H.0 – P0H.7: | A8 – A15 | AD8 - AD15 | | | | | | | | | | | | | | | | | | | | | |
| P1L.0 - P1L.7 P1H.0 - P1H.7 | 118-125 128-135 | I/O | <p>Two 8-bit bidirectional I/O ports P1L and P1H, bit-wise programmable for input or output via direction bit. Programming an I/O pin as input forces the corresponding output driver to high impedance state. PORT1 is used as the 16-bit address bus (A) in demultiplexed bus modes: if at least BUSCONx is configured such the demultiplexed mode is selected, the pins of PORT1 are not available for general purpose I/O function. The input threshold of Port 1 is selectable (TTL or CMOS).</p> <p>The pins of P1L also serve as the additional (up to 8) analog input channels for the A/D converter, where P1L.x equals ANy (Analog input channel y, where $y = x + 16$). This additional function have higher priority on demultiplexed bus function. The following PORT1 pins have alternate functions:</p> | | | | | | | | | | | | | | | | | | | | |
| | 132 | I | P1H.4 | CC24IO | CAPCOM2: CC24 capture input | | | | | | | | | | | | | | | | | | |
| | 133 | I | P1H.5 | CC25IO | CAPCOM2: CC25 capture input | | | | | | | | | | | | | | | | | | |
| | 134 | I | P1H.6 | CC26IO | CAPCOM2: CC26 capture input | | | | | | | | | | | | | | | | | | |
| | 135 | I | P1H.7 | CC27IO | CAPCOM2: CC27 capture input | | | | | | | | | | | | | | | | | | |

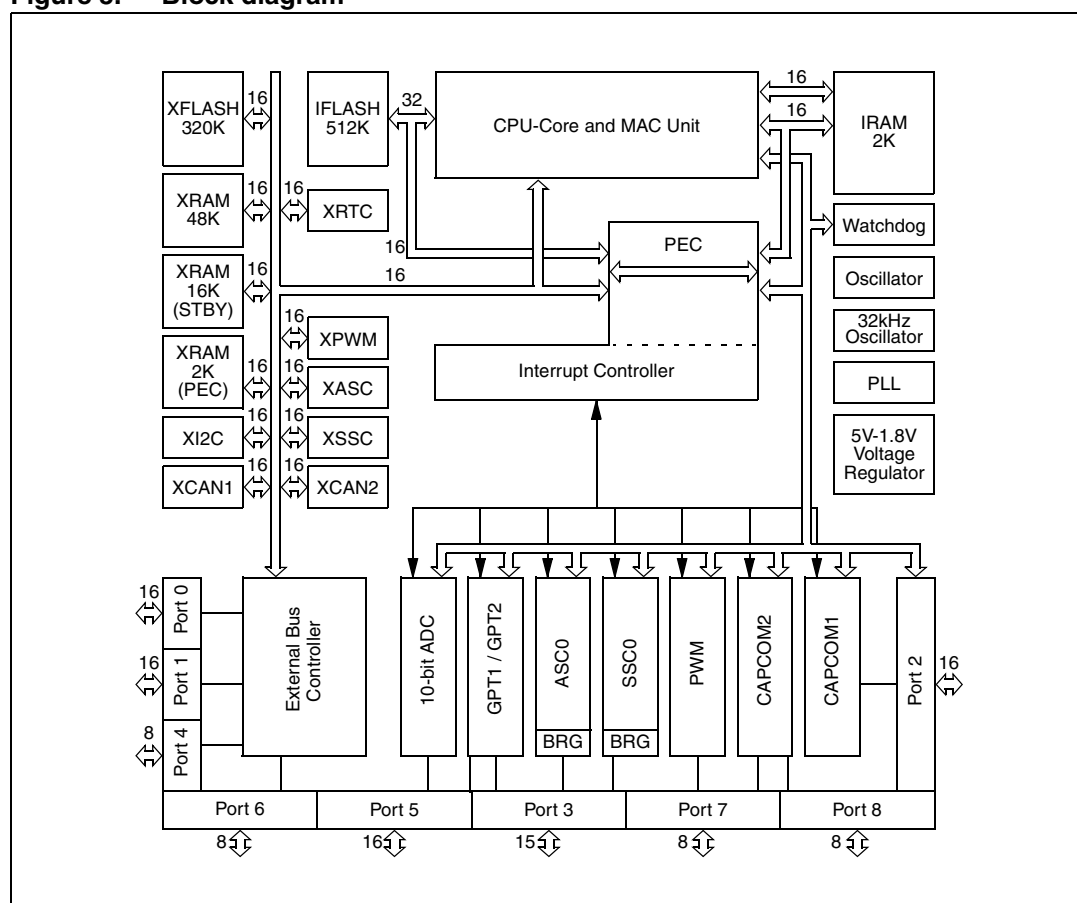
Table 1. Pin description (continued)

| Symbol | Pin | Type | Function | |
|----------------------------|---|------|---|--|
| XTAL1 | 138 | I | XTAL1 | Main oscillator amplifier circuit and/or external clock input. |
| XTAL2 | 137 | O | XTAL2 | Main oscillator amplifier circuit output. |
| | | | To clock the device from an external source, drive XTAL1 while leaving XTAL2 unconnected. Minimum and maximum high / low and rise / fall times specified in the AC Characteristics must be observed. | |
| XTAL3 | 143 | I | XTAL3 | 32 kHz oscillator amplifier circuit input |
| XTAL4 | 144 | O | XTAL4 | 32 kHz oscillator amplifier circuit output |
| | | | When 32 kHz oscillator amplifier is not used, to avoid spurious consumption, XTAL3 shall be tied to ground while XTAL4 shall be left open. Besides, bit OFF32 in RTCCON register shall be set. 32 kHz oscillator can only be driven by an external crystal, and not by a different clock source. | |
| $\overline{\text{RSTIN}}$ | 140 | I | Reset Input with CMOS Schmitt-Trigger characteristics. A low level at this pin for a specified duration while the oscillator is running resets the ST10F276. An internal pull-up resistor permits power-on reset using only a capacitor connected to V_{SS} . In bidirectional reset mode (enabled by setting bit BDRSTEN in SYSCON register), the $\overline{\text{RSTIN}}$ line is pulled low for the duration of the internal reset sequence. | |
| $\overline{\text{RSTOUT}}$ | 141 | O | Internal Reset Indication Output. This pin is driven to a low level during hardware, software or watchdog timer reset. $\overline{\text{RSTOUT}}$ remains low until the EINIT (end of initialization) instruction is executed. | |
| $\overline{\text{NMI}}$ | 142 | I | Non-Maskable Interrupt Input. A high to low transition at this pin causes the CPU to vector to the NMI trap routine. If bit PWDCFG = '0' in SYSCON register, when the PWRDN (power down) instruction is executed, the $\overline{\text{NMI}}$ pin must be low in order to force the ST10F276 to go into power down mode. If $\overline{\text{NMI}}$ is high and PWDCFG = '0', when PWRDN is executed, the part will continue to run in normal mode. If not used, pin $\overline{\text{NMI}}$ should be pulled high externally. | |
| V_{AREF} | 37 | - | A/D converter reference voltage and analog supply | |
| V_{AGND} | 38 | - | A/D converter reference and analog ground | |
| RPD | 84 | - | Timing pin for the return from interruptible power down mode and synchronous / asynchronous reset selection. | |
| V_{DD} | 17, 46, 72, 82, 93, 109, 126, 136 | - | Digital supply voltage = + 5V during normal operation, idle and power down modes. It can be turned off when Stand-by RAM mode is selected. | |
| V_{SS} | 18, 45, 55, 71, 83, 94, 110, 127, 139 | - | Digital ground | |
| V_{18} | 56 | - | 1.8V decoupling pin: a decoupling capacitor (typical value of 10nF, max 100nF) must be connected between this pin and nearest V_{SS} pin. | |

3 Functional description

The architecture of the ST10F276 combines advantages of both RISC and CISC processors and an advanced peripheral subsystem. The block diagram gives an overview of the different on-chip components and the high bandwidth internal bus structure of the ST10F276.

Figure 3. Block diagram

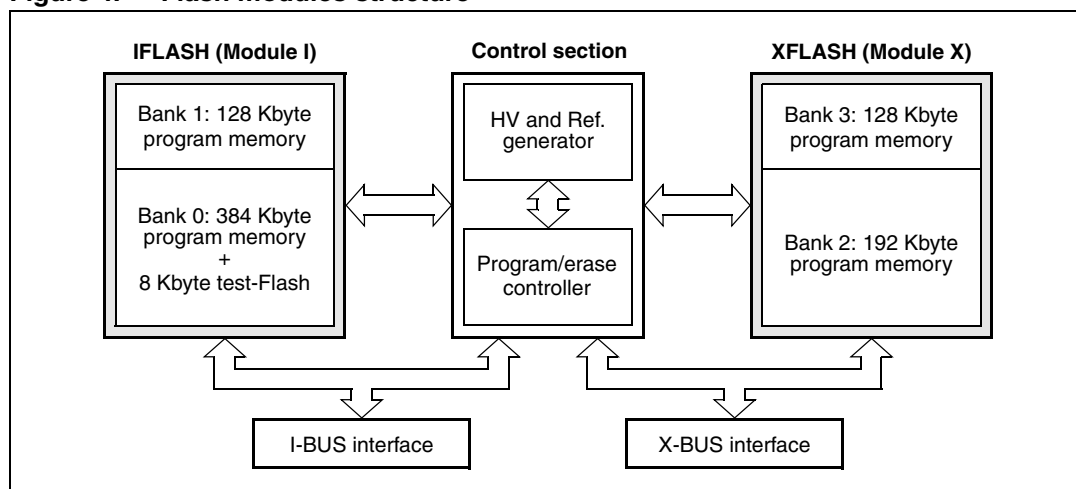


4 Internal Flash memory

4.1 Overview

The on-chip Flash is composed by two matrix modules each one containing one array divided in two banks that can be read and modified independently one of the other: one bank can be read while another bank is under modification.

Figure 4. Flash modules structure



The write operations of the 4 banks are managed by an embedded Flash program/erase controller (FPEC). The high voltages needed for program/erase operations are internally generated.

The data bus is 32-bit wide. Due to ST10 core architecture limitation, only the first 512 Kbytes are accessed at 32-bit (internal Flash bus, see I-BUS), while the remaining 320 Kbytes are accessed at 16-bit (see X-BUS).

4.2 Functional description

4.2.1 Structure

The following table shows the address space reserved to the Flash module.

Table 2. Flash modules absolute mapping

| Description | Addresses | Size |
|--|------------------------|-----------|
| IFLASH sectors | 0x00 0000 to 0x08 FFFF | 512 Kbyte |
| XFLASH sectors | 0x09 0000 to 0x0D FFFF | 320 Kbyte |
| Registers and Flash internal reserved area | 0x0E 0000 to 0x0E FFFF | 64 Kbyte |

4.2.2 Modules structure

The IFLASH module is composed by 2 banks. Bank 0 contains 384 Kbyte of program memory divided in 10 sectors. Bank 0 contains also a reserved sector named test-Flash. Bank 1 contains 128 Kbyte of program memory or parameter divided in 2 sectors (64 Kbyte each).

The XFLASH module is composed by 2 banks as well. Bank 2 contains 192 Kbyte of Program Memory divided in 3 sectors. Bank 3 contains 128 Kbyte of program memory or parameter divided in 2 sectors (64 Kbyte each).

Addresses from 0x0E 0000 to 0x0E FFFF are reserved for the control register interface and other internal service memory space used by the Flash program/erase controller.

The following tables show the memory mapping of the Flash when it is accessed in read mode ([Table 3](#)), and when accessed in write or erase mode ([Table 2](#)): note that with this second mapping, the first three banks are remapped into code segment 1 (same as obtained when setting bit ROMS1 in SYSCON register).

Table 3. Flash modules sectorization (read operations)

| Bank | Description | Addresses | Size | ST10 bus size |
|------|-----------------------|---------------------------|-------|----------------|
| B0 | Bank 0 Flash 0 (B0F0) | 0x0000 0000 - 0x0000 1FFF | 8 KB | 32-bit (I-BUS) |
| | Bank 0 Flash 1 (B0F1) | 0x0000 2000 - 0x0000 3FFF | 8 KB | |
| | Bank 0 Flash 2 (B0F2) | 0x0000 4000 - 0x0000 5FFF | 8 KB | |
| | Bank 0 Flash 3 (B0F3) | 0x0000 6000 - 0x0000 7FFF | 8 KB | |
| | Bank 0 Flash 4 (B0F4) | 0x0001 8000 - 0x0001 FFFF | 32 KB | |
| | Bank 0 Flash 5 (B0F5) | 0x0002 0000 - 0x0002 FFFF | 64 KB | |
| | Bank 0 Flash 6 (B0F6) | 0x0003 0000 - 0x0003 FFFF | 64 KB | |
| | Bank 0 Flash 7 (B0F7) | 0x0004 0000 - 0x0004 FFFF | 64 KB | |
| | Bank 0 Flash 8 (B0F8) | 0x0005 0000 - 0x0005 FFFF | 64 KB | |
| | Bank 0 Flash 9 (B0F9) | 0x0006 0000 - 0x0006 FFFF | 64 KB | |
| B1 | Bank 1 Flash 0 (B1F0) | 0x0007 0000 - 0x0007 FFFF | 64 KB | 32-bit (I-BUS) |
| | Bank 1 Flash 1 (B1F1) | 0x0008 0000 - 0x0008 FFFF | 64 KB | |
| B2 | Bank 2 Flash 0 (B2F0) | 0x0009 0000 - 0x0009 FFFF | 64 KB | |
| | Bank 2 Flash 1 (B2F1) | 0x000A 0000 - 0x000A FFFF | 64 KB | |
| | Bank 2 Flash 2 (B2F2) | 0x000B 0000 - 0x000B FFFF | 64 KB | |
| B3 | Bank 3 Flash 0 (B3F0) | 0x000C 0000 - 0x000C FFFF | 64 KB | |
| | Bank 3 Flash 1 (B3F1) | 0x000D 0000 - 0x000D FFFF | 64 KB | |
| | | | | 16-bit (X-BUS) |

Table 4. Flash modules sectorization (write operations or with roms1='1')

| Bank | Description | Addresses | Size | ST10 Bus size |
|------|--------------------------|---------------------------|-------|----------------|
| B0 | Bank 0 Test-Flash (B0TF) | 0x0000 0000 - 0x0000 1FFF | 8 KB | 32-bit (I-BUS) |
| | Bank 0 Flash 0 (B0F0) | 0x0001 0000 - 0x0001 1FFF | 8 KB | |
| | Bank 0 Flash 1 (B0F1) | 0x0001 2000 - 0x0001 3FFF | 8 KB | |
| | Bank 0 Flash 2 (B0F2) | 0x0001 4000 - 0x0001 5FFF | 8 KB | |
| | Bank 0 Flash 3 (B0F3) | 0x0001 6000 - 0x0001 7FFF | 8 KB | |
| | Bank 0 Flash 4 (B0F4) | 0x0001 8000 - 0x0001 FFFF | 32 KB | |
| | Bank 0 Flash 5 (B0F5) | 0x0002 0000 - 0x0002 FFFF | 64 KB | |
| | Bank 0 Flash 6 (B0F6) | 0x0003 0000 - 0x0003 FFFF | 64 KB | |
| | Bank 0 Flash 7 (B0F7) | 0x0004 0000 - 0x0004 FFFF | 64 KB | |
| | Bank 0 Flash 8 (B0F8) | 0x0005 0000 - 0x0005 FFFF | 64 KB | |
| | Bank 0 Flash 9 (B0F9) | 0x0006 0000 - 0x0006 FFFF | 64 KB | |
| B1 | Bank 1 Flash 0 (B1F0) | 0x0007 0000 - 0x0007 FFFF | 64 KB | 16-bit (X-BUS) |
| | Bank 1 Flash 1 (B1F1) | 0x0008 0000 - 0x0008 FFFF | 64 KB | |
| B2 | Bank 2 Flash 0 (B2F0) | 0x0009 0000 - 0x0009 FFFF | 64 KB | |
| | Bank 2 Flash 1 (B2F1) | 0x000A 0000 - 0x000A FFFF | 64 KB | |
| | Bank 2 Flash 2 (B2F2) | 0x000B 0000 - 0x000B FFFF | 64 KB | |
| B3 | Bank 3 Flash 0 (B3F0) | 0x000C 0000 - 0x000C FFFF | 64 KB | |
| | Bank 3 Flash 1 (B3F1) | 0x000D 0000 - 0x000D FFFF | 64 KB | |

The table above refers to the configuration when bit ROMS1 of SYSCON register is set. When Bootstrap mode is entered:

- Test-Flash is seen and available for code fetches (address 00'0000h)
- User IFlash is only available for read and write accesses
- Write accesses must be made with addresses starting in segment 1 from 01'0000h, whatever ROMS1 bit in SYSCON value
- Read accesses are made in segment 0 or in segment 1 depending of ROMS1 value.

In Bootstrap mode, by default ROMS1 = 0, so the first 32KBytes of IFlash are mapped in segment 0.

Example: In default configuration, to program address 0, user must put the value 01'0000h in the FARL and FARH registers, but to verify the content of the address 0 a read to 00'0000h must be performed.

Table 5 shows the control register interface composition: this set of registers can be addressed by the CPU.

Table 5. Control register interface

| Bank | Description | Addresses | Size | ST10 bus size |
|---------|---|---------------------------|--------|----------------|
| FCR1-0 | Flash control registers 1-0 | 0x000E 0000 - 0x000E 0007 | 8 byte | 16-bit (X-BUS) |
| FDR1-0 | Flash data registers 1-0 | 0x000E 0008 - 0x000E 000F | 8 byte | |
| FAR | Flash address registers | 0x000E 0010 - 0x000E 0013 | 4 byte | |
| FER | Flash error register | 0x000E 0014 - 0x000E 0015 | 2 byte | |
| FNWVPXR | Flash non volatile protection X register | 0x000E DFB0 - 0x000E DFB3 | 4 byte | |
| FNWVPIR | Flash non volatile protection I register | 0x000E DFB4 - 0x000E DFB7 | 4 byte | |
| FNVAPR0 | Flash non volatile access protection register 0 | 0x000E DFB8 - 0x000E DFB9 | 2 byte | |
| FNVAPR1 | Flash non volatile access protection register 1 | 0x000E DFBC - 0x000E DFBF | 4 byte | |
| XFICR | XFlash interface control register | 0x000E E000 - 0x000E E001 | 2 byte | |

4.2.3 Low power mode

The Flash modules are automatically switched off executing PWRDN instruction. The consumption is drastically reduced, but exiting this state can require a long time (t_{PD}).

Note: Recovery time from Power Down mode for the Flash modules is anyway shorter than the main oscillator start-up time. To avoid any problem in restarting to fetch code from the Flash, it is important to size properly the external circuit on RPD pin.

Power-off Flash mode is entered only at the end of the eventually running Flash write operation.

4.3 Write operation

The Flash modules have one single register interface mapped in the memory space of the XFlash module (0x0E 0000 to 0x0E 0013). All the operations are enabled through four 16-bit control registers: Flash Control Register 1-0 High/Low (FCR1H/L-FCR0H/L). Eight other 16-bit registers are used to store Flash Address and Data for Program operations (FARH/L and FDR1H/L-FDR0H/L) and Write Operation Error flags (FERH/L). All registers are accessible with 8 and 16-bit instructions (since mapped on ST10 XBUS).

Note: Before accessing the XFlash module (and consequently also the Flash register to be used for program/erasing operations), bit XFLASHEN in XPERCON register and bit XPEN in SYSCON register shall be set.

The 4 Banks have their own dedicated sense amplifiers, so that any Bank can be read while any other Bank is written. However simultaneous write operations ("write" means either Program or Erase) on different Banks are forbidden: when there is a write operation on going (Program or Erase) anywhere in the Flash, no other write operation can be performed.

During a Flash write operation any attempt to read the bank under modification will output invalid data (software trap 009Bh). This means that the Flash Bank is not fetchable when a write operation is active: the write operation commands must be executed from another

Bank, or from the other module or again from another memory (internal RAM or external memory).

Note: During a Write operation, when bit LOCK of FCR0 is set, it is forbidden to write into the Flash Control Registers.

4.3.1 Power supply drop

If during a write operation the internal low voltage supply drops below a certain internal voltage threshold, any write operation running is suddenly interrupted and the modules are reset to Read mode. At following Power-on, an interrupted Flash write operation must be repeated.

4.4 Registers description

4.4.1 Flash control register 0 low

The Flash control register 0 low (FCR0L) together with the Flash control register 0 high (FCR0H) is used to enable and to monitor all the write operations for both the Flash modules. The user has no access in write mode to the test-Flash (B0TF). Besides, test-Flash block is seen by the user in Bootstrap mode only.

| FCR0L (0x0E 0000) | | | | | | FCR | | | | | | Reset value: 0000h | | | |
|-------------------|----|----|----|----|----|-----|---|---|------|------|------|--------------------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| reserved | | | | | | | | | BSY1 | BSY0 | LOCK | res. | BSY3 | BSY2 | res. |
| | | | | | | | | | R | R | R | | R | R | |

Table 6. Flash control register 0 low

| Bit | Function |
|----------|---|
| BSY(3:2) | <p>Bank 3:2 Busy (XFLASH)</p> <p>These bits indicate that a write operation is running on the corresponding Bank of XFLASH. They are automatically set when bit WMS is set. Setting Protection operation sets bit BSY2 (since protection registers are in the Block B2). When these bits are set every read access to the corresponding Bank will output invalid data (software trap 009Bh), while every write access to the Bank will be ignored. At the end of the write operation or during a Program or Erase Suspend these bits are automatically reset and the Bank returns to read mode. After a Program or Erase Resume these bits are automatically set again.</p> |

Table 6. Flash control register 0 low (continued)

| Bit | Function |
|----------|--|
| LOCK | Flash registers access locked When this bit is set, it means that the access to the Flash Control Registers FCR0H/-FCR1H/L, FDR0H/L-FDR1H/L, FARH/L and FER is locked by the FPEC: any read access to the registers will output invalid data (software trap 009Bh) and any write access will be ineffective. LOCK bit is automatically set when the Flash bit WMS is set. This is the only bit the user can always access to detect the status of the Flash: once it is found low, the rest of FCR0L and all the other Flash registers are accessible by the user as well. Note that FER content can be read when LOCK is low, but its content is updated only when also BSY bits are reset. |
| BSY(1:0) | Bank 1:0 Busy (IFLASH) These bits indicate that a write operation is running in the corresponding Bank of IFLASH. They are automatically set when bit WMS is set. When these bits are set every read access to the corresponding Bank will output invalid data (software trap 009Bh), while every write access to the Bank will be ignored. At the end of the write operation or during a Program or Erase Suspend these bits are automatically reset and the Bank returns to read mode. After a Program or Erase Resume these bits are automatically set again. |

4.4.2 Flash control register 0 high

The Flash control register 0 high (FCR0H) together with the Flash control register 0 Low (FCR0L) is used to enable and to monitor all the write operations for both the Flash modules. The user has no access in write mode to the Test-Flash (BOTF). Besides, test-Flash block is seen by the user in Bootstrap mode only.

| FCR0H (0x0E 0002) | | | | | FCR | | | | | Reset value: 0000h | | | | | |
|-------------------|------|-----|------|-----|----------|---|-----|------|----------|--------------------|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| WMS | SUSP | WPG | DWPG | SER | Reserved | | SPR | SMOD | Reserved | | | | | | |
| RW | RW | RW | RW | RW | | | RW | RW | | | | | | | |

Table 7. Flash control register 0 high

| Bit | Function |
|------|--|
| SMOD | Select module If this bit is reset, the Write Operation is performed on XFLASH Module; if this bit is set, the Write Operation is performed on IFLASH Module. SMOD bit is automatically reset at the end of the Write operation. |
| SPR | Set protection This bit must be set to select the Set Protection operation. The Set Protection operation allows to program 0s in place of 1s in the Flash Non Volatile Protection Registers. The Flash Address in which to program must be written in the FARH/L registers, while the Flash Data to be programmed must be written in the FDR0H/L before starting the execution by setting bit WMS. A sequence error is flagged by bit SEQER of FER if the address written in FARH/L is not in the range 0x0EDFB0-0x0EDFBF. SPR bit is automatically reset at the end of the Set Protection operation. |

Table 7. Flash control register 0 high (continued)

| Bit | Function |
|------|---|
| SER | <p>Sector erase</p> <p>This bit must be set to select the Sector Erase operation in the Flash modules. The Sector Erase operation allows to erase all the Flash locations to 0xFF. From 1 to all the sectors of the same Bank (excluded Test-Flash for Bank B0) can be selected to be erased through bits BxFy of FCR1H/L registers before starting the execution by setting bit WMS. It is not necessary to pre-program the sectors to 0x00, because this is done automatically. SER bit is automatically reset at the end of the Sector Erase operation.</p> |
| DWPG | <p>Double word program</p> <p>This bit must be set to select the Double Word (64 bits) Program operation in the Flash modules. The Double Word Program operation allows to program 0s in place of 1s. The Flash Address in which to program (aligned with even words) must be written in the FARH/L registers, while the 2 Flash Data to be programmed must be written in the FDR0H/L registers (even word) and FDR1H/L registers (odd word) before starting the execution by setting bit WMS. DWPG bit is automatically reset at the end of the Double Word Program operation.</p> |
| WPG | <p>Word program</p> <p>This bit must be set to select the Word (32 bits) Program operation in the Flash modules. The Word Program operation allows to program 0s in place of 1s. The Flash Address to be programmed must be written in the FARH/L registers, while the Flash Data to be programmed must be written in the FDR0H/L registers before starting the execution by setting bit WMS. WPG bit is automatically reset at the end of the Word Program operation.</p> |
| SUSP | <p>Suspend</p> <p>This bit must be set to suspend the current Program (Word or Double Word) or Sector Erase operation in order to read data in one of the Sectors of the Bank under modification or to program data in another Bank. The Suspend operation resets the Flash Bank to normal read mode (automatically resetting bits BSYx). When in Program Suspend, the two Flash modules accept only the following operations: Read and Program Resume. When in Erase Suspend the modules accept only the following operations: Read, Erase Resume and Program (Word or Double Word; Program operations cannot be suspended during Erase Suspend). To resume the suspended operation, the WMS bit must be set again, together with the selection bit corresponding to the operation to resume (WPG, DWPG, SER).</p> <p>Note: It is forbidden to start a new Write operation with bit SUSP already set.</p> |
| WMS | <p>Write mode start</p> <p>This bit must be set to start every write operation in the Flash modules. At the end of the write operation or during a Suspend, this bit is automatically reset. To resume a suspended operation, this bit must be set again. It is forbidden to set this bit if bit ERR or FER is high (the operation is not accepted). It is also forbidden to start a new write (program or erase) operation (by setting WMS high) when bit SUSP of FCR0 is high. Resetting this bit by software has no effect.</p> |

4.4.3 Flash control register 1 low

The Flash control register 1 low (FCR1L), together with Flash control register 1 high (FCR1H), is used to select the sectors to erase, or during any write operation to monitor the status of each sector of the module selected by SMOD bit of FCR0H. First diagram shows FCR1L meaning when SMOD=0; the second one when SMOD=1.

| FCR1L (0x0E 0004) SMOD=0 | | | | | | | | | | FCR | | | | Reset value: 0000h | | | |
|--------------------------|----|----|----|----|----|---|---|---|---|-----|---|---|------|--------------------|------|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| Reserved | | | | | | | | | | | | | B2F2 | B2F1 | B2F0 | | |
| | | | | | | | | | | | | | RS | RS | RS | | |

| FCR1L (0x0E 0004) SMOD=1 | | | | | | | | | | FCR | | | | Reset value: 0000h | | | |
|--------------------------|----|----|----|----|----|------|------|------|------|------|------|------|------|--------------------|------|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| Reserved | | | | | | B0F9 | B0F8 | B0F7 | B0F6 | B0F5 | B0F4 | B0F3 | B0F2 | B0F1 | B0F0 | | |
| | | | | | | RS | RS | RS | RS | RS | RS | RS | RS | RS | RS | | |

Table 8. Flash control register 1 low

| Bit | Function |
|---------------------------------|---|
| SMOD=0 (XFLASH selected) | |
| B2F(2:0) | Bank 2 XFLASH sector 2:0 status These bits must be set during a Sector Erase operation to select the sectors to erase in bank 2. Besides, during any erase operation, these bits are automatically set and give the status of the 3 sectors of bank 2 (B2F2-B2F0). The meaning of B2Fy bit for sector y of bank 2 is given by the next Table 10. These bits are automatically reset at the end of a write operation if no errors are detected. |
| SMOD=1 (IFLASH selected) | |
| B0F(9:0) | Bank 0 IFLASH sector 9:0 status These bits must be set during a Sector Erase operation to select the sectors to erase in bank 0. Besides, during any erase operation, these bits are automatically set and give the status of the 10 sectors of bank 0 (B0F9-B0F0). The meaning of B0Fy bit for sector y of bank 0 is given by the next Table 10 . These bits are automatically reset at the end of a Write operation if no errors are detected. |

4.4.4 Flash control register 1 high

The Flash control register 1 high (FCR1H), together with Flash control register 1 low (FCR1L), is used to select the sectors to erase, or during any write operation to monitor the status of each sector and each bank of the module selected by SMOD bit of FCR0H. First diagram shows FCR1H meaning when SMOD=0; the second one when SMOD=1.

| FCR1H (0x0E 0006) SMOD=0 | | | | | | FCR | | | | | | Reset value: 0000h | | | |
|--------------------------|----|----|----|----|----|-----|-----|----------|---|---|---|--------------------|---|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| reserved | | | | | | B3S | B2S | reserved | | | | | | B3F1 | B3F0 |
| | | | | | | RS | RS | | | | | | | RS | RS |

| FCR1H (0x0E 0006) SMOD=1 | | | | | | FCR | | | | | | Reset value: 0000h | | | |
|--------------------------|----|----|----|----|----|-----|-----|----------|---|---|---|--------------------|---|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| reserved | | | | | | B1S | B0S | reserved | | | | | | B1F1 | B1F0 |
| - | | | | | | RS | RS | | | | | | | RS | RS |

Table 9. Flash control register 1 high

| Bit | Function |
|---------------------------------|--|
| SMOD=0 (XFLASH selected) | |
| B3F(1:0) | Bank 3 XFLASH sector 1:0 status During any erase operation, these bits are automatically set and give the status of the 2 sectors of bank 3 (B3F1-B3F0). The meaning of B3Fy bit for sector y of bank 1 is given by the next Table 10 . These bits are automatically reset at the end of a erase operation if no errors are detected. |
| B(3:2)S | Bank 3-2 status (XFLASH) During any erase operation, these bits are automatically modified and give the status of the 2 Banks (B3-B2). The meaning of BxS bit for bank x is given in the next Table 10 . These bits are automatically reset at the end of a erase operation if no errors are detected. |
| SMOD=1 (IFLASH selected) | |
| B1F(1:0) | Bank 1 IFLASH sector 1:0 status During any erase operation, these bits are automatically set and give the status of the 2 sectors of bank 1 (B1F1-B1F0). The meaning of B1Fy bit for sector y of bank 1 is given by the next Table 10 . These bits are automatically reset at the end of a erase operation if no errors are detected. |
| B(1:0)S | Bank 1-0 status (IFLASH) During any erase operation, these bits are automatically modified and give the status of the 2 banks (B1-B0). The meaning of BxS bit for bank x is given in the next Table 10 . These bits are automatically reset at the end of a erase operation if no errors are detected. |

During any erase operation, these bits are automatically set and give the status of the 2 sectors of Bank 1 (B1F1-B1F0). The meaning of B1Fy bit for sector y of bank 1 is given by the next [Table 10](#). These bits are automatically reset at the end of a erase operation if no errors are detected.

Table 10. Banks (BxS) and sectors (BxFy) status bits meaning

| ERR | SUSP | BxS = 1 meaning | BxFy = 1 meaning |
|-----|------|---------------------------|---------------------------------------|
| 1 | - | Erase error in bank x | Erase error in sector y of bank x |
| 0 | 1 | Erase suspended in bank x | Erase suspended in sector y of bank x |
| 0 | 0 | Don't care | Don't care |

4.4.5 Flash data register 0 low

The Flash address registers (FARH/L) and the Flash data registers (FDR1H/L-FDR0H/L) are used during the program operations to store Flash Address in which to program and data to program.

| FDR0L (0x0E 0008) | | | | | | | FCR | | | | Reset value: FFFFh | | | | | |
|-------------------|-------|-------|-------|-------|-------|------|------|------|------|------|--------------------|------|------|------|------|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| DIN15 | DIN14 | DIN13 | DIN12 | DIN11 | DIN10 | DIN9 | DIN8 | DIN7 | DIN6 | DIN5 | DIN4 | DIN3 | DIN2 | DIN1 | DIN0 | |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | |

Table 11. Flash data register 0 low

| Bit | Function |
|-----------|---|
| DIN(15:0) | Data input 15:0 These bits must be written with the data to program the Flash with the following operations: word program (32-bit), double word program (64-bit) and set protection. |

4.4.6 Flash data register 0 high

| FDR0H (0x0E 000A) | | | | | | | FCR | | | | Reset value: FFFFh | | | | | |
|-------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--------------------|-------|-------|-------|-------|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| DIN31 | DIN30 | DIN29 | DIN28 | DIN27 | DIN26 | DIN25 | DIN24 | DIN23 | DIN22 | DIN21 | DIN20 | DIN19 | DIN18 | DIN17 | DIN16 | |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | |

Table 12. Flash data register 0 high

| Bit | Function |
|------------|--|
| DIN(31:16) | Data input 31:16 These bits must be written with the data to program the Flash with the following operations: word program (32-bit), double word program (64-bit) and set protection. |

4.4.7 Flash data register 1 low

| FDR1L (0x0E 000C) | | | | | | | FCR | | | | Reset value: FFFFh | | | | | |
|-------------------|-------|-------|-------|-------|-------|------|------|------|------|------|--------------------|------|------|------|------|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| DIN15 | DIN14 | DIN13 | DIN12 | DIN11 | DIN10 | DIN9 | DIN8 | DIN7 | DIN6 | DIN5 | DIN4 | DIN3 | DIN2 | DIN1 | DIN0 | |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | |

Table 13. Flash data register 1 low

| Bit | Function |
|-----------|---|
| DIN(15:0) | Data Input 15:0 These bits must be written with the Data to program the Flash with the following operations: Word Program (32-bit), Double Word Program (64-bit) and Set Protection. |

4.4.8 Flash data register 1 high

| FDR1H (0x0E 000E) | | | | | | | FCR | | | | | | | Reset value: FFFFh | |
|-------------------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--------------------|-------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DIN31 | DIN30 | DIN29 | DIN28 | DIN27 | DIN26 | DIN25 | DIN24 | DIN23 | DIN22 | DIN21 | DIN20 | DIN19 | DIN18 | DIN17 | DIN16 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

Table 14. Flash data register 1 high

| Bit | Function |
|------------|--|
| DIN(31:16) | Data input 31:16 These bits must be written with the data to program the Flash with the following operations: word program (32-bit), double word program (64-bit) and set protection. |

4.4.9 Flash address register low

| FARL (0x0E 0010) | | | | | | | FCR | | | | | | | Reset value: 0000h | |
|------------------|-------|-------|-------|-------|-------|------|------|------|------|------|------|------|------|--------------------|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADD15 | ADD14 | ADD13 | ADD12 | ADD11 | ADD10 | ADD9 | ADD8 | ADD7 | ADD6 | ADD5 | ADD4 | ADD3 | ADD2 | reserved | |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | | |

Table 15. Flash address register low

| Bit | Function |
|-----------|---|
| ADD(15:2) | Address 15:2 These bits must be written with the address of the Flash location to program in the following operations: word program (32-bit) and double word program (64-bit). In double word program bit ADD2 must be written to '0'. |

4.4.10 Flash address register high

| FARH (0x0E 0012) | | | | | | | | | FCR | | | | | Reset value: 0000h | | | | |
|------------------|----|----|----|----|----|---|---|---|-----|---|-------|-------|-------|--------------------|-------|--|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| reserved | | | | | | | | | | | ADD20 | ADD19 | ADD18 | ADD17 | ADD16 | | | |
| | | | | | | | | | | | RW | RW | RW | RW | RW | | | |

Table 16. Flash address register high

| Bit | Function |
|------------|--|
| ADD(20:16) | Address 20:16 These bits must be written with the address of the Flash location to program in the following operations: word program and double word program. |

4.4.11 Flash error register

Flash error register, as well as all the other Flash registers, can be properly read only once LOCK bit of register FCR0L is low. Nevertheless, its content is updated when also BSY bits are reset as well; for this reason, it is definitively meaningful reading FER register content only when LOCK bit and all BSY bits are cleared.

| FER (0xE 0014h) | | | | | | | | | FCR | | | | | Reset value: 0000h | | | | |
|-----------------|----|----|----|----|----|---|-----|-------|-------|----------|---|------|------|--------------------|-----|--|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| reserved | | | | | | | WPF | RESER | SEQER | reserved | | 10ER | PGER | ERER | ERR | | | |
| | | | | | | | RC | RC | RC | | | RC | RC | RC | RC | | | |

Table 17. Flash error register

| Bit | Function |
|------|---|
| ERR | Write error This bit is automatically set when an error occurs during a Flash write operation or when a bad write operation setup is done. Once the error has been discovered and understood, ERR bit must be software reset. |
| ERER | Erase error This bit is automatically set when an erase error occurs during a Flash write operation. This error is due to a real failure of a Flash cell, that can no more be erased. This kind of error is fatal and the sector where it occurred must be discarded. This bit has to be software reset. |
| PGER | Program error This bit is automatically set when a program error occurs during a Flash write operation. This error is due to a real failure of a Flash cell, that can no more be programmed. The word where this error occurred must be discarded. This bit has to be software reset. |
| 10ER | 1 over 0 error This bit is automatically set when trying to program at 1 bits previously set at 0 (this does not happen when programming the protection bits). This error is not due to a failure of the Flash cell, but only flags that the desired data has not been written. This bit has to be software reset. |

Table 17. Flash error register (continued)

| Bit | Function |
|-------|---|
| SEQR | Sequence error This bit is automatically set when the control registers (FCR1H/L-FCR0H/L, FARH/L, FDR1H/L-FDR0H/L) are not correctly filled to execute a valid write operation. In this case no write operation is executed. This bit has to be software reset. |
| RESER | Resume error This bit is automatically set when a suspended program or erase operation is not resumed correctly due to a protocol error. In this case the suspended operation is aborted. This bit has to be software reset. |
| WPF | Write protection flag This bit is automatically set when trying to program or erase in a sector write protected. In case of multiple sector erase, the not protected sectors are erased, while the protected sectors are not erased and bit WPF is set. This bit has to be software reset. |

4.4.12 XFlash interface control register

This register is used to configure the XFLASH interface behaviour on the XBUS. It allows to set the number of wait states introduced on the XBUS before the internal $\overline{\text{READY}}$ signal is given to the ST10 bus master.

| XFICR (0xE E000h) | | | | | | | | | | XBUS | | | | Reset value: 000Fh | | | |
|-------------------|----|----|----|----|----|---|---|---|---|------|---|-----|-----|--------------------|-----|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| reserved | | | | | | | | | | | | WS3 | WS2 | WS1 | WS0 | | |
| | | | | | | | | | | | | RW | RW | RW | RW | | |

Table 18. XFlash interface control register

| Bit | Function |
|---------|---|
| WS(3:0) | Wait state setting These three bits are the binary coding of the number of wait states introduced by the XFLASH interface through the XBUS internal $\overline{\text{READY}}$ signal. Default value after reset is 1111, that is the up to 15 wait states are set. The following recommendations for the ST10F276 are hereafter reported: For $f_{\text{CPU}} > 40\text{MHz}$ Wait-StateWS(3:0) = '0001' For $f_{\text{CPU}} \leq 40\text{MHz}$ Wait-StateWS(3:0) = '0000' |

4.5 Protection strategy

The protection bits are stored in Non Volatile Flash cells inside XFLASH module, that are read once at reset and stored in 7 Volatile registers. Before they are read from the Non Volatile cells, all the available protections are forced active during reset.

The protections can be programmed using the Set Protection operation (see Flash Control Registers paragraph), that can be executed from all the internal or external memories except from the Flash Bank B2.

Two kind of protections are available: write protections to avoid unwanted writings and access protections to avoid piracy. In next paragraphs all different level of protections are shown, and architecture limitations are highlighted as well.

4.5.1 Protection registers

The 7 Non Volatile Protection Registers are one time programmable for the user.

Four registers (FNVWPXRL/H-FNVWPIRL/H) are used to store the Write Protection fuses respectively for each sector of the XFLASH Module (see X) and IFLASH module (see I). The other three Registers (FNVAPR0 and FNVAPR1L/H) are used to store the Access Protection fuses (common to both Flash modules even though with some limitations).

4.5.2 Flash non volatile write protection X register low

| FNVWPXRL (0x0E DFB0) | | | | | | NVR | | | | | | Delivery value: FFFFh | | | |
|----------------------|----------|----|----|----|----|-----|---|---|---|---|---|-----------------------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| W2PPR | reserved | | | | | | | | | | | | W2P2 | W2P1 | W2P0 |
| RW | | | | | | | | | | | | | RW | RW | RW |

Table 19. Flash non volatile write protection X register low

| Bit | Function |
|----------|---|
| W2P(2:0) | Write Protection Bank 2 sectors 2-0 (XFLASH) These bits, if programmed at 0, disable any write access to the sectors of Bank 2 (XFLASH). |
| W2PPR | Write Protection Bank 2 Non Volatile cells This bit, if programmed at 0, disables any write access to the Non Volatile cells of Bank 2. Since these Non Volatile cells are dedicated to Protection registers, once W2PPR bit is set, the configuration of protection setting is frozen, and can only be modified executing a Temporary Write Unprotection operation. |

4.5.3 Flash non volatile write protection X register high

| FNVWPXRH (0x0E DF B2) | | | | | | | | | | NVR | | | | Delivery value: FFFFh | | | |
|-----------------------|----|----|----|----|----|---|---|---|---|-----|---|---|---|-----------------------|------|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| reserved | | | | | | | | | | | | | | W3P1 | W3P0 | | |
| | | | | | | | | | | | | | | RW | RW | | |

Table 20. Flash non volatile write protection X register high

| Bit | Function |
|----------|---|
| W3P(1:0) | Write Protection Bank 3 / Sectors 1-0 (XFLASH) These bits, if programmed at 0, disable any write access to the sectors of Bank 3 (XFLASH). |

4.5.4 Flash non volatile write protection I register low

| FNVWPIRL (0x0E DFB4) | | | | | | NVR | | | | | | Delivery value: FFFFh | | | |
|----------------------|----|----|----|----|----|------|------|------|------|------|------|-----------------------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| reserved | | | | | | W0P9 | W0P8 | W0P7 | W0P6 | W0P5 | W0P4 | W0P3 | W0P2 | W0P1 | W0P0 |
| | | | | | | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

Table 21. Flash non volatile write protection I register low

| Bit | Function |
|----------|---|
| W0P(9:0) | Write Protection Bank 0 / Sectors 9-0 (IFLASH) These bits, if programmed at 0, disable any write access to the sectors of Bank 0 (IFLASH). |

4.5.5 Flash non volatile write protection I register high

| FNVWPIRH (0x0E DFB6) | | | | | | | | | | NVR | | | | Delivery value: FFFFh | | | |
|----------------------|----|----|----|----|----|---|---|---|---|-----|---|---|---|-----------------------|------|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| reserved | | | | | | | | | | | | | | W1P1 | W1P0 | | |
| | | | | | | | | | | | | | | RW | RW | | |

Table 22. Flash non volatile write protection I register high

| Bit | Function |
|----------|---|
| W1P(1:0) | Write Protection Bank 1 / Sectors 1-0 (IFLASH) These bits, if programmed at 0, disable any write access to the sectors of Bank 1 (IFLASH). |

4.5.6 Flash non volatile access protection register 0

Due to ST10 architecture, the XFLASH is seen as external memory: this made impossible to access protect it from real external memory or internal RAM.

| FNVAPR0 (0x0E DFB8) | | | | | | NVR | | | | | | Delivery value: ACFFh | | | |
|---------------------|----|----|----|----|----|-----|---|---|---|---|---|-----------------------|------|------|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| reserved | | | | | | | | | | | | | DBGP | ACCP | |
| | | | | | | | | | | | | | RW | RW | |

Table 23. Flash non volatile access protection register 0

| Bit | Function |
|------|---|
| ACCP | Access Protection This bit, if programmed at 0, disables any access (read/write) to data mapped inside IFlash Module address space, unless the current instruction is fetched from one of the two Flash modules. |
| DBGP | Debug Protection This bit, if erased at 1, allows to by-pass all the protections using the Debug features through the Test Interface. If programmed at 0, on the contrary, all the debug features, the Test Interface and all the Flash Test Modes are disabled. Even STMicroelectronics will not be able to access the device to run any eventual failure analysis. |

4.5.7 Flash non volatile access protection register 1 low

| FNVAPR1L (0x0E DFBC) | | | | | | NVR | | | | | | Delivery value: FFFFh | | | |
|----------------------|-------|-------|-------|-------|-------|------|------|------|------|------|------|-----------------------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PDS15 | PDS14 | PDS13 | PDS12 | PDS11 | PDS10 | PDS9 | PDS8 | PDS7 | PDS6 | PDS5 | PDS4 | PDS3 | PDS2 | PDS1 | PDS0 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

Table 24. Flash non volatile access protection register 1 low

| Bit | Function |
|-----------|--|
| PDS(15:0) | Protections Disable 15-0 If bit PDSx is programmed at 0 and bit PENx is erased at 1, the action of bit ACCP is disabled. Bit PDS0 can be programmed at 0 only if bits DBGP and ACCP have already been programmed at 0. Bit PDSx can be programmed at 0 only if bit PENx-1 has already been programmed at 0. |

4.5.8 Flash non volatile access protection register 1 high

| FNVAPR1H (0x0E DFBE) | | | | | | NVR | | | | | | Delivery value: FFFFh | | | |
|----------------------|-------|-------|-------|-------|-------|------|------|------|------|------|------|-----------------------|------|------|------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PEN15 | PEN14 | PEN13 | PEN12 | PEN11 | PEN10 | PEN9 | PEN8 | PEN7 | PEN6 | PEN5 | PEN4 | PEN3 | PEN2 | PEN1 | PEN0 |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

Table 25. Flash non volatile access protection register 1 high

| Bit | Function |
|---------|--|
| PEN15-0 | Protections Enable 15-0 If bit PENx is programmed at 0 and bit PDSx+1 is erased at 1, the action of bit ACCP is enabled again. Bit PENx can be programmed at 0 only if bit PDSx has already been programmed at 0. |

4.5.9 Access protection

The Flash modules have one level of access protection (access to data both in Reading and Writing): if bit ACCP of FNVAPR0 is programmed at 0, the IFlash module become access protected: data in the IFlash module can be read/written only if the current execution is from the IFlash module itself.

Protection can be permanently disabled by programming bit PDS0 of FNVAPR1H, in order to analyze rejects. Allowing PDS0 bit programming only when ACCP bit is programmed, guarantees that only an execution from the Flash itself can disable the protections.

Protection can be permanently enabled again by programming bit PEN0 of FNVAPR1L. The action to disable and enable again Access Protections in a permanent way can be executed a maximum of 16 times.

Trying to write into the access protected Flash from internal RAM will be unsuccessful. Trying to read into the access protected Flash from internal RAM will output a dummy data.

When the Flash module is protected in access, also the data access through PEC of a peripheral is forbidden. To read/write data in PEC mode from/to a protected Bank, first it is necessary to temporary unprotect the Flash module.

Due to ST10 architecture, the XFLASH is seen as external memory: this makes impossible to access protect it from real external memory or internal RAM. In the following table a summary of all levels of possible Access protection is reported: in particular, supposing to enable all possible access protections, when fetching from a memory as listed in the first column, what is possible and what is not possible to do (see column headers) is shown in the table.

Table 26. Summary of access protection level

| | Read IFLASH / Jump to IFLASH | Read XFLASH /Jump to XFLASH | Read FLASH Registers | Write FLASH Registers |
|----------------------|------------------------------------|-----------------------------------|-------------------------|--------------------------|
| Fetching from IFLASH | Yes / Yes | Yes / Yes | Yes | Yes |
| Fetching from XFLASH | No / Yes | Yes / Yes | Yes | No |
| Fetching from IRAM | No / Yes | Yes / Yes | Yes | No |

Table 26. Summary of access protection level

| | Read IFLASH / Jump to IFLASH | Read XFLASH /Jump to XFLASH | Read FLASH Registers | Write FLASH Registers |
|----------------------------------|------------------------------------|-----------------------------------|-------------------------|--------------------------|
| Fetching from XRAM | No / Yes | Yes / Yes | Yes | No |
| Fetching from External Memory | No / Yes | Yes / Yes | Yes | No |

4.5.10 Write protection

The Flash modules have one level of Write Protections: each Sector of each Bank of each Flash Module can be Software Write Protected by programming at 0 the related bit WyPx of FNVWPXRH/L-FNVWPIRH/L registers.

4.5.11 Temporary unprotection

Bits WyPx of FNVWPXRH/L-FNVWPIRH/L can be temporary unprotected by executing the Set Protection operation and writing 1 into these bits.

Bit ACCP can be temporary unprotected by executing the Set Protection operation and writing 1 into these bits, but only if these write instructions are executed from the Flash Modules.

To restore the write and access protection bits it is necessary to reset the microcontroller or to execute a Set Protection operation and write 0 into the desired bits.

It is not necessary to temporary unprotect an access protected Flash in order to update the code: it is, in fact, sufficient to execute the updating instructions from another Flash Bank.

In reality, when a temporary unprotection operation is executed, the corresponding volatile register is written to 1, while the non volatile registers bits previously written to 0 (for a protection set operation), will continue to maintain the 0. For this reason, the User software must be in charge to track the current protection status (for instance using a specific RAM area), it is not possible to deduce it by reading the non volatile register content (a temporary unprotection cannot be detected).

4.6 Write operation examples

In the following, examples for each kind of Flash write operation are presented.

Word program

Example: 32-bit Word Program of data 0xAAAAAAAA at address 0x0C5554 in XFLASH Module.

```
FCR0H|= 0x2000; /*Set WPG in FCR0H*/
FARL = 0x5554; /*Load Add in FARL*/
FARH = 0x000C; /*Load Add in FARH*/
FDR0L = 0xAAAA; /*Load Data in FDR0L*/
FDR0H = 0xAAAA; /*Load Data in FDR0H*/
FCR0H|= 0x8000; /*Operation start*/
```

Double word program

Example: Double Word Program (64-bit) of data 0x55AA55AA at address 0x095558 and data 0xAA55AA55 at address 0x09555C in IFLASH Module.

```
FCR0H|= 0x1080; /*Set DWPG, SMOD*/
FARL = 0x5558; /*Load Add in FARL*/
FARH = 0x0009; /*Load Add in FARH*/
FDR0L = 0x55AA; /*Load Data in FDR0L*/
FDR0H = 0x55AA; /*Load Data in FDR0H*/
FDR1L = 0xAA55; /*Load Data in FDR1L*/
FDR1H = 0xAA55; /*Load Data in FDR1H*/
FCR0H|= 0x8000; /*Operation start*/
```

Double Word Program is always performed on the Double Word aligned on a even Word: bit ADD2 of FARL is ignored.

Sector erase

Example: Sector Erase of sectors B3F1 and B3F0 of Bank 3 in XFLASH Module.

```
FCR0H|= 0x0800; /*Set SER in FCR0H*/
FCR1H|= 0x0003; /*Set B3F1, B3F0*/
FCR0H|= 0x8000; /*Operation start*/
```

Suspend and resume

Word Program, Double Word Program, and Sector Erase operations can be suspended in the following way:

```
FCR0H|= 0x4000; /*Set SUSP in FCR0H*/
```

Then the operation can be resumed in the following way:

```
FCR0H|= 0x0800; /*Set SER in FCR0H*/
FCR0H|= 0x8000; /*Operation resume*/
```

Before resuming a suspended Erase, FCR1H/FCR1L must be read to check if the Erase is already completed (FCR1H = FCR1L = 0x0000 if Erase is complete). Original setup of Select Operation bits in FCR0H/L must be restored before the operation resume, otherwise the operation is aborted and bit RESER of FER is set.

Erase suspend, program and resume

A Sector Erase operation can be suspended in order to program (Word or Double Word) another Sector.

Example: Sector Erase of sector B3F1 of Bank 3 in XFLASH Module.

```
FCR0H|= 0x0800; /*Set SER in FCR0H*/
FCR1H|= 0x0002; /*Set B3F1*/
FCR0H|= 0x8000; /*Operation start*/
```

Example: Sector Erase Suspend.

```
FCR0H|= 0x4000; /*Set SUSP in FCR0H*/
do
    /*Loop to wait for LOCK=0 and WMS=0*/
{tmp1 = FCR0L;
 tmp2 = FCR0H;
 } while ((tmp1 && 0x0010) || (tmp2 && 0x8000));
```

Example: Word Program of data 0x5555AAAA at address 0x0C5554 in XFLASH module.

```
FCR0H&= 0xBFFF; /*Rst SUSP in FCR0H*/
FCR0H|= 0x2000; /*Set WPG in FCR0H*/
FARL = 0x5554; /*Load Add in FARL*/
FARH = 0x000C; /*Load Add in FARH*/
FDR0L = 0xAAAA; /*Load Data in FDR0L*/
FDR0H = 0x5555; /*Load Data in FDR0H*/
FCR0H|= 0x8000; /*Operation start*/
```

Once the Program operation is finished, the Erase operation can be resumed in the following way:

```
FCR0H|= 0x0800; /*Set SER in FCR0H*/
FCR0H|= 0x8000; /*Operation resume*/
```

Notice that during the Program Operation in Erase suspend, bits SER and SUSP are low. A Word or Double Word Program during Erase Suspend cannot be suspended.

To summarize:

- A Sector Erase can be suspended by setting SUSP bit
- To perform a Word Program operation during Erase Suspend, firstly bits SUSP and SER must be reset, then bit WPG and WMS can be set
- To resume the Sector Erase operation bit SER must be set again
- In any case it is forbidden to start any write operation with SUSP bit already set

Set protection

Example 1: Enable Write Protection of sectors B0F3-0 of Bank 0 in IFLASH module.

```
FCR0H|= 0x0100; /*Set SPR in FCR0H*/
FARL = 0xDFB4; /*Load Add of register FNVWPIRL in FARL*/
FARH = 0x000E; /*Load Add of register FNVWPIRL in FARH*/
FDR0L = 0xFFFF0; /*Load Data in FDR0L*/
FDR0H = 0xFFFF; /*Load Data in FDR0H*/
FCR0H|= 0x8000; /*Operation start*/
```

Notice that bit SMOD of FCR0H must not be set, since Write Protection bits of IFLASH Module are stored in Test-Flash (XFLASH Module).

Example 2: Enable Access and Debug Protection.

```
FCR0H|= 0x0100; /*Set SPR in FCR0H*/
FARL = 0xDFB8; /*Load Add of register FNVAPR0 in FARL*/
FARH = 0x000E; /*Load Add of register FNVAPR0 in FARH*/
FDR0L = 0xFFFFC; /*Load Data in FDR0L*/
FCR0H|= 0x8000; /*Operation start*/
```

Example 3: Disable in a permanent way Access and Debug Protection.

```
FCR0H|= 0x0100; /*Set SPR in FCR0H*/
FARL = 0xDFBC; /*Load Add of register FNVAPR1L in FARL*/
FARH = 0x000E; /*Load Add of register FNVAPR1L in FARH*/
FDR0L = 0xFFFFE; /*Load Data in FDR0L for clearing PDS0*/
FCR0H|= 0x8000; /*Operation start*/
```

Example 4: Enable again in a permanent way Access and Debug Protection, after having disabled them.

```
FCR0H|= 0x0100; /*Set SPR in FCR0H*/
FARL = 0xDFBC; /*Load Add register FNVAPR1H in FARL*/
FARH = 0x000E; /*Load Add register FNVAPR1H in FARH*/
FDR0H = 0xFFFFE; /*Load Data in FDR0H for clearing PEN0*/
FCR0H|= 0x8000; /*Operation start*/
```

Disable and re-enable of Access and Debug Protection in a permanent way (as shown by examples 3 and 4) can be done for a maximum of 16 times.

4.7 Write operation summary

In general, each write operation is started through a sequence of 3 steps:

1. The first instruction is used to select the desired operation by setting its corresponding selection bit in the Flash Control Register 0. This instruction is also used to select in which Flash Module to apply the Write Operation (by setting/resetting bit SMOD).
2. The second step is the definition of the Address and Data for programming or the Sectors or Banks to erase.
3. The last instruction is used to start the write operation, by setting the start bit WMS in the FCR0.

Once selected, but not yet started, one operation can be canceled by resetting the operation selection bit.

A summary of the available Flash Module Write Operations are shown in the following [Table 27](#).

Table 27. Flash write operations

| Operation | Select bit | Address and Data | Start bit |
|------------------------------|------------|---|-----------|
| Word Program (32-bit) | WPG | FARL/FARH FDR0L/FDR0H | WMS |
| Double Word Program (64-bit) | DWPG | FARL/FARH FDR0L/FDR0H FDR1L/FDR1H | WMS |
| Sector Erase | SER | FCR1L/FCR1H | WMS |

Table 27. Flash write operations (continued)

| Operation | Select bit | Address and Data | Start bit |
|-----------------------|------------|------------------|-----------|
| Set Protection | SPR | FDR0L/FDR0H | WMS |
| Program/Erase Suspend | SUSP | None | None |

5 Bootstrap loader

ST10F276 implements innovative boot capabilities in order to

- support a user defined bootstrap (see *Alternate bootstrap loader*);
- support bootstrap via UART or bootstrap via CAN for the standard bootstrap.

5.1 Selection among user-code, standard or alternate bootstrap

The selection among user-code, standard bootstrap or alternate bootstrap is made by special combinations on Port0L[5...4] during the time the reset configuration is latched from Port0.

The alternate boot mode is triggered with a special combination set on Port0L[5...4]. Those signals, as other configuration signals, are latched on the rising edge of $\overline{\text{RSTIN}}$ pin.

The alternate boot function is divided in two functional parts (which are independent from each other):

Part 1: Selection of reset sequence according to the Port0 configuration: User mode and alternate mode signatures

- Decoding of reset configuration (P0L.5 = 1, P0L.4 = 1) selects the normal mode and the user Flash to be mapped from address 00'0000h.
- Decoding of reset configuration (P0L.5 = 1, P0L.4 = 0) selects ST10 standard bootstrap mode (Test-Flash is active and overlaps user Flash for code fetches from address 00'0000h; user Flash is active and available for read and program).
- Decoding of reset configuration (P0L.5 = 0, P0L.4 = 1) activates new verifications to select which bootstrap software to execute:
 - if the user mode signature in the user Flash is programmed correctly, then a software reset sequence is selected and the user code is executed;
 - if the user mode signature is not programmed correctly but the alternate mode signature in the user Flash is programmed correctly, then the alternate boot mode is selected;
 - if both the user and the alternate mode signatures are not programmed correctly in the user Flash, then the user key location is read again. Its value will determine the behavior of the selected bootstrap loader.

Part 2: Running of user selected reset sequence

- Standard bootstrap loader: Jump to a predefined memory location in Test-Flash (controlled by ST)
- Alternate boot mode: Jump to address 09'0000h
- Selective bootstrap loader: Jump to a predefined location in Test-Flash (controlled by ST) and check which communication channel is selected
- User code: Make a software reset and jump to 00'0000h

Table 28. ST10F276 boot mode selection

| P0.5 | P0.4 | ST10 decoding |
|------|------|---|
| 1 | 1 | User mode: User Flash mapped at 00'0000h |
| 1 | 0 | Standard bootstrap loader: User Flash mapped from 00'0000h; code fetches redirected to Test-Flash at 00'0000h |
| 0 | 1 | Alternate boot mode: Flash mapping depends on signatures integrity check |
| 0 | 0 | Reserved |

5.2 Standard bootstrap loader

The built-in bootstrap loader of the ST10F276 provides a mechanism to load the startup program, which is executed after reset, via the serial interface. In this case no external (ROM) memory or an internal ROM is required for the initialization code starting at location 00'0000_H. The bootstrap loader moves code/data into the IRAM but it is also possible to transfer data via the serial interface into an external RAM using a second level loader routine. ROM memory (internal or external) is not necessary. However, it may be used to provide lookup tables or may provide “core-code”, that is, a set of general purpose subroutines, such as for I/O operations, number crunching or system initialization.

The Bootstrap Loader can load

- the complete application software into ROMless systems,
- temporary software into complete systems for testing or calibration,
- a programming routine for Flash devices.

The BSL mechanism may be used for standard system start-up as well as for only special occasions like system maintenance (firmware update) or end-of-line programming or testing.

5.2.1 Entering the standard bootstrap loader

As with the old ST10 bootstrap mode, the ST10F276 enters BSL mode if pin P0L.4 is sampled low at the end of a hardware reset. In this case, the built-in bootstrap loader is activated independently of the selected bus mode. The bootstrap loader code is stored in a special Test-Flash; no part of the standard Flash memory area is required for this.

After entering BSL mode and the respective initialization, the ST10F276 scans the RxD0 line and the CAN1_RxD line to receive either a valid dominant bit from the CAN interface or a start condition from the UART line.

Start condition on UART RxD: The ST10F276 starts the standard bootstrap loader. This bootstrap loader is identical to other ST10 devices (Examples: ST10F269, ST10F168). See paragraph 5.3 for details.

Valid dominant bit on CAN1 RxD: The ST10F276 starts bootstrapping via CAN1; the bootstrapping method is new and is described in the next paragraph 5.4. The following Figure 5 shows the program flow of the new bootstrap loader. It clearly illustrates how the new functionalities are implemented:

- UART: UART has priority over CAN after a falling edge on CAN1_RxD until the first valid rising edge on CAN1_RxD;
- CAN: Pulses on CAN1_RxD shorter than 20*CPU-cycles are filtered.

5.2.2 ST10 configuration in BSL

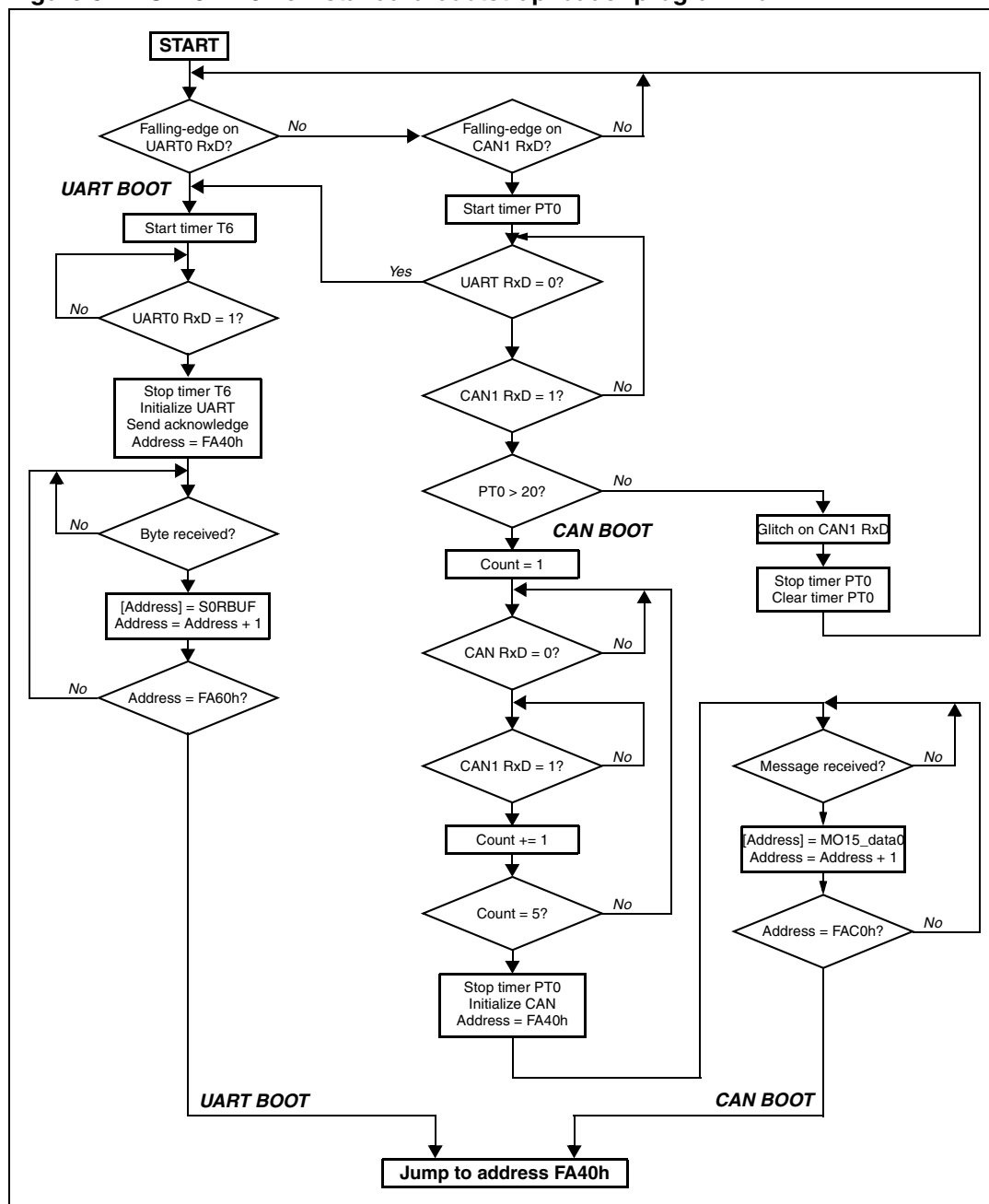
When the ST10F276 has entered BSL mode, the configuration shown in [Table 29](#) is automatically set (values that deviate from the normal reset values are marked in **bold**).

Table 29. ST10 configuration in BSL mode

| Function or register | Access | Notes |
|------------------------------|--|--|
| Watchdog Timer | Disabled | |
| Register SYSCON | 0404_H ⁽¹⁾ | XPEN bit set for Bootstrap via CAN or Alternate Boot Mode |
| Context Pointer CP | FA00 _H | |
| Register STKUN | FC00 _H | |
| Stack Pointer SP | FA40 _H | |
| Register STKOV | FA00 _H | |
| Register BUSCON0 | acc. to startup config. ⁽²⁾ | |
| Register S0CON | 8011 _H | Initialized only if Bootstrap via UART |
| Register S0BG | acc. to '00' byte | Initialized only if Bootstrap via UART |
| P3.10 / TXD0 | '1' | Initialized only if Bootstrap via UART |
| DP3.10 | '1' | Initialized only if Bootstrap via UART |
| CAN1 Status/Control Register | 0000 _H | Initialized only if Bootstrap via CAN |
| CAN1 Bit Timing Register | acc. to '0' frame | Initialized only if Bootstrap via CAN |
| XPERCON | 042D _H | XRAM1-2, XFlash, CAN1 and XMISC enabled. Initialized only if Bootstrap via CAN |
| P4.6 / CAN1_TxD | '1' | Initialized only if Bootstrap via CAN |
| DP4.6 | '1' | Initialized only if Bootstrap via CAN |

1. In Bootstrap modes (standard or alternate) ROMEN, bit 10 of SYSCON, is always set regardless of \overline{EA} pin level. BYTDIS, bit 9 of SYSCON, is set according to data bus width selection via Port0 configuration.
2. BUSCON0 is initialized with 0000h, external bus disabled, if pin \overline{EA} is high during reset. If pin \overline{EA} is low during reset, BUSACT0, bit 10, and ALECTL0, bit 9, are set enabling the external bus with lengthened ALE signal. BTYP field, bit 7 and 6, is set according to Port0 configuration.

Figure 5. ST10F276 new standard bootstrap loader program flow



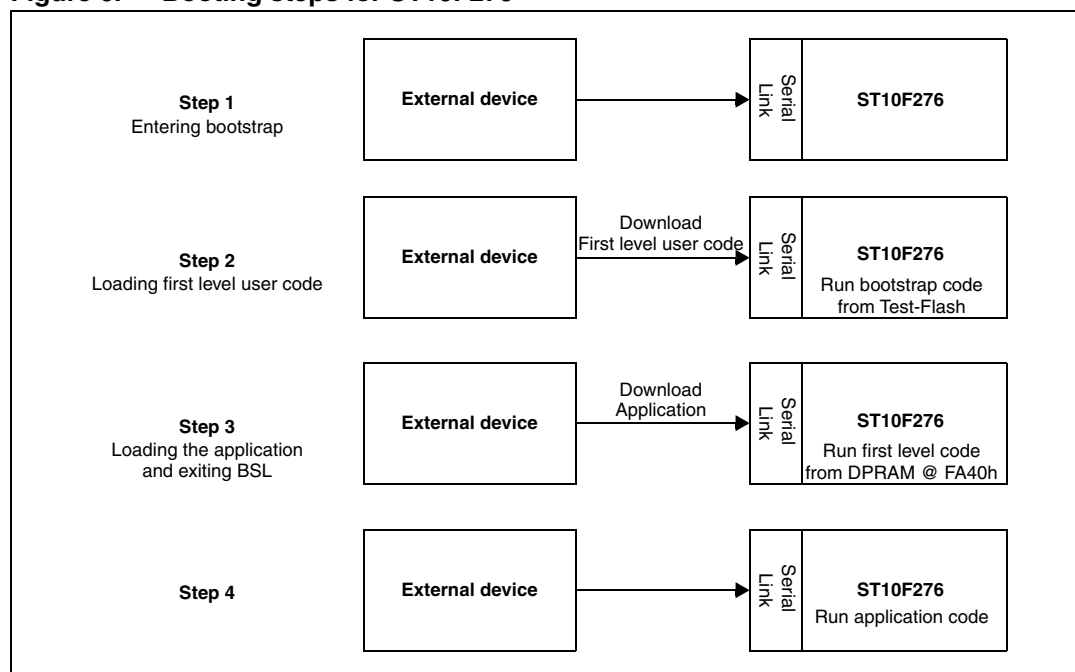
Other than after a normal reset the watchdog timer is disabled, so the bootstrap loading sequence is not time limited. Depending on the selected serial link (UART0 or CAN1), pin TxD0 or CAN1_TxD is configured as output, so the ST10F276 can return the acknowledge byte. Even if the internal IFLASH is enabled, a code cannot be executed from it.

5.2.3 Booting steps

As [Figure 6](#) shows, booting ST10F276 with the boot loader code occurs in a minimum of four steps:

1. The ST10F276 is reset with P0L.4 low.
2. The internal new bootstrap code runs on the ST10 and a first level user code is downloaded from the external device, via the selected serial link (UART0 or CAN1). The bootstrap code is contained in the ST10F276 Test-Flash and is automatically run when ST10F276 is reset with P0L.4 low. After loading a preselected number of bytes, ST10F276 begins executing the downloaded program.
3. The first level user code runs on ST10F276. Typically, this first level user code is another loader that downloads the application software into the ST10F276.
4. The loaded application software is now running.

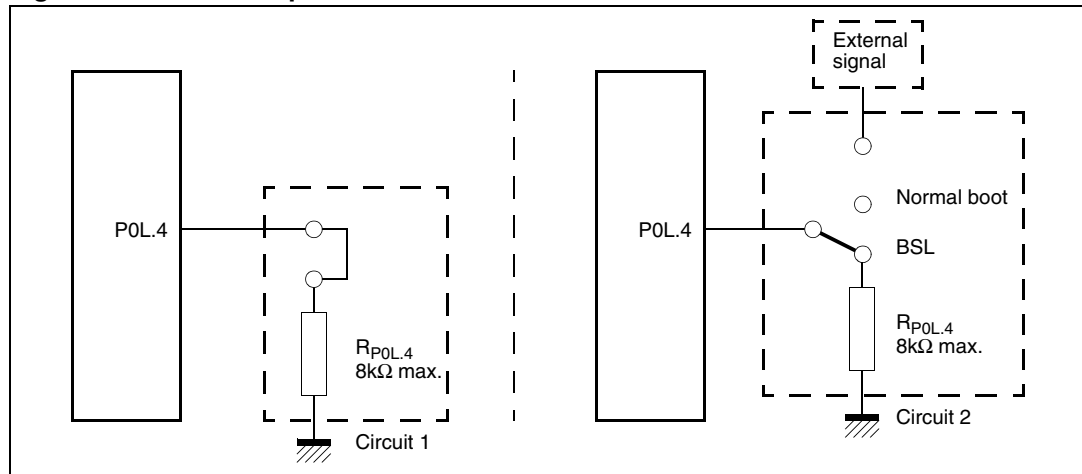
Figure 6. Booting steps for ST10F276



5.2.4 Hardware to activate BSL

The hardware that activates the BSL during reset may be a simple pull-down resistor on P0L.4 for systems that use this feature at every hardware reset. For systems that use the bootstrap loader only temporarily, it may be preferable to use a switchable solution (via jumper or an external signal).

Note: CAN alternate function on Port4 lines is not activated if the user has selected eight address segments (Port4 pins have three functions: I/O port, address segment and CAN). Boot via CAN requires that four or less address segments are selected.

Figure 7. Hardware provisions to activate the BSL

5.2.5 Memory configuration in bootstrap loader mode

The configuration (that is, the accessibility) of the ST10F276's memory areas after reset in Bootstrap Loader mode differs from the standard case. Pin \overline{EA} is evaluated when BSL mode is selected to enable or to not enable the external bus:

- If $\overline{EA} = 1$, the external bus is disabled (BUSACT0 = 0 in BUSCON0 register);
- If $\overline{EA} = 0$, the external bus is enabled (BUSACT0 = 1 in BUSCON0 register).

Moreover, while in BSL mode, accesses to the internal IFLASH area are partially redirected:

- All code accesses are made from the special Test-Flash seen in the range 00'0000h to 00'01FFFh;
- User IFLASH is only available for read and write accesses (Test-Flash cannot be read or written);
- Write accesses must be made with addresses starting in segment 1 from 01'0000h, regardless of the value of ROMS1 bit in SYSCON register;
- Read accesses are made in segment 0 or in segment 1 depending on the ROMS1 value;
- In BSL mode, by default, ROMS1 = 0, so the first 32 Kbytes of IFlash are mapped in segment 0.

Example:

In default configuration, to program address 0, the user must put the value 01'0000h in the FARL and FARH registers but to verify the content of the address 0 a read to 00'0000h must be performed.

Figure 8. Memory configuration after reset

| | 16 Mbytes | 16 Mbytes | 16 Mbytes |
|-------------------------------------|--------------------------|--------------------------|--------------------------|
| | | | |
| BSL mode active | Yes (P0L.4 = '0') | Yes (P0L.4 = '0') | No (P0L.4 = '1') |
| \overline{EA} pin | High | Low | According to application |
| Code fetch from internal FLASH area | Test-FLASH access | Test-FLASH access | User IFLASH access |
| Data fetch from internal FLASH area | User IFLASH access | User IFLASH access | User IFLASH access |

Note: As long as ST10F276 is in BSL, the user's software should not try to execute code from the internal IFlash, as the fetches are redirected to the Test-Flash.

5.2.6 Loading the start-up code

After the serial link initialization sequence (see following chapters), the BSL enters a loop to receive 32 bytes (boot via UART) or 128 bytes (boot via CAN).

These bytes are stored sequentially into ST10F276 Dual-Port RAM from location 00'FA40h.

To execute the loaded code, the BSL then jumps to location 00'FA40h. The bootstrap sequence running from the Test-Flash is now terminated; however, the microcontroller remains in BSL mode.

Most probably, the initially loaded routine, being the first level user code, will load additional code and data. This first level user code may use the pre-initialized interface (UART or CAN) to receive data and a second level of code, and store it in arbitrary user-defined locations.

This second level of code may be

- the final application code
- another, more sophisticated, loader routine that adds a transmission protocol to enhance the integrity of the loaded code or data
- a code sequence to change the system configuration and enable the bus interface to store the received data into external memory

In all cases, the ST10F276 still runs in BSL mode, that is, with the watchdog timer disabled and limited access to the internal IFLASH area.

5.2.7 Exiting bootstrap loader mode

To execute a program in normal mode, the BSL mode must first be terminated. The ST10F276 exits BSL mode at a software reset (level on P0L.4 is ignored) or a hardware reset (P0L.4 must be high in this case). After the reset, the ST10F276 starts executing from location 00'0000_H of the internal Flash (User Flash) or the external memory, as programmed via pin \overline{EA} .

Note: If a bidirectional Software Reset is executed and external memory boot is selected ($\overline{EA} = 0$), a degeneration of the Software Reset event into a Hardware Reset can occur (refer to section for details). This implies that P0L.4 becomes transparent, so to exit from Bootstrap mode it would be necessary to release pin P0L.4 (it is no longer ignored).

5.2.8 Hardware requirements

Although the new bootstrap loader is designed to be compatible with the old bootstrap loader, there are a few hardware requirements relative to the new bootstrap loader:

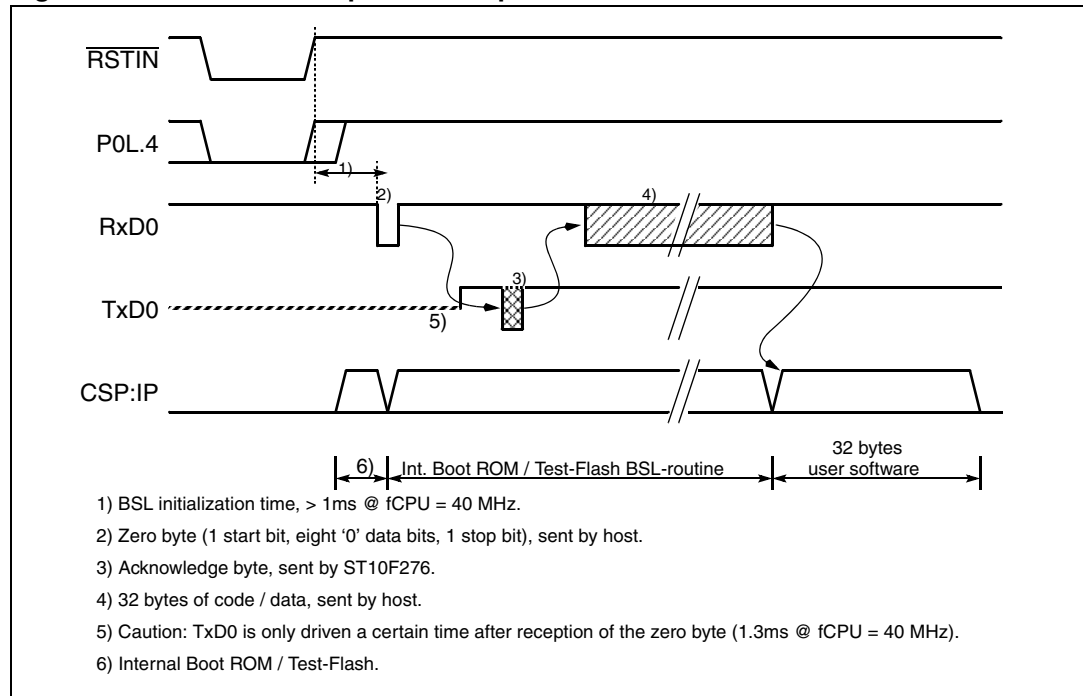
- External Bus configuration: Must have four or less segment address lines (keep CAN I/Os available);
- Usage of CAN pins (P4.5 and P4.6): Even in bootstrap via UART, P4.5 (CAN1_RxD) can be used as Port input but not as output. The pin P4.6 (CAN1_TxD) can be used as input or output.
- Level on UART RxD and CAN1_RxD during the bootstrap phase (see Figure 6 - Step 2): Must be 1 (external pull-ups recommended).

5.3 Standard bootstrap with UART (RS232 or K-Line)

5.3.1 Features

ST10F276 bootstrap via UART has the same overall behavior as the old ST10 bootstrap via UART:

- Same bootstrapping steps;
- Same bootstrap method: Analyze the timing of a predefined byte, send back an acknowledge byte, load a fixed number of bytes and run;
- Same functionalities: Boot with different crystals and PLL ratios.

Figure 9. UART bootstrap loader sequence

5.3.2 Entering bootstrap via UART

The ST10F276 enters BSL mode if pin P0L.4 is sampled low at the end of a hardware reset. In this case, the built-in bootstrap loader is activated independently of the selected bus mode. The bootstrap loader code is stored in a special Test-Flash; no part of the standard mask ROM or Flash memory area is required for this.

After entering BSL mode and the respective initialization, the ST10F276 scans the RxD0 line to receive a zero byte, that is, 1 start bit, eight '0' data bits and 1 stop bit. From the duration of this zero byte, it calculates the corresponding baud rate factor with respect to the current CPU clock, initializes the serial interface ASC0 accordingly and switches pin TxD0 to output. Using this baud rate, an acknowledge byte is returned to the host that provides the loaded data.

The acknowledge byte is **D5h** for the ST10F276.

5.3.3 ST10 Configuration in UART BSL (RS232 or K-Line)

When the ST10F276 enters BSL mode on UART, the configuration shown in [Table 30](#) is automatically set (values that deviate from the normal reset values are marked in **bold**).

Table 30. ST10 configuration in UART BSL mode (RS232 or K-line)

| Function or register | Access | Notes |
|----------------------|--|--|
| Watchdog timer | Disabled | |
| Register SYSCON | 0400_H ⁽¹⁾ | |
| Context Pointer CP | FA00 _H | |
| Register STKUN | FA00 _H | |
| Stack Pointer SP | FA40 _H | |
| Register STKOV | FC00 _H | |
| Register BUSCON0 | acc. to startup config. ⁽²⁾ | |
| Register S0CON | 8011 _H | Initialized only if Bootstrap via UART |
| Register S0BG | acc. to '00' byte | Initialized only if Bootstrap via UART |
| P3.10 / TXD0 | '1' | Initialized only if Bootstrap via UART |
| DP3.10 | '1' | Initialized only if Bootstrap via UART |

1. In Bootstrap modes (standard or alternate) ROMEN, bit 10 of SYSCON, is always set regardless of \overline{EA} pin level. BYTDIS, bit 9 of SYSCON, is set according to data bus width selection via Port0 configuration.
2. BUSCON0 is initialized with 0000h, external bus disabled, if pin \overline{EA} is high during reset. If pin \overline{EA} is low during reset, BUSACT0, bit 10, and ALECTL0, bit 9, are set enabling the external bus with lengthened ALE signal. BTYP field, bit 7 and 6, is set according to Port0 configuration.

Other than after a normal reset, the watchdog timer is disabled, so the bootstrap loading sequence is not time limited. Pin TxD0 is configured as output, so the ST10F276 can return the acknowledge byte. Even if the internal IFLASH is enabled, a code cannot be executed from it.

5.3.4 Loading the start-up code

After sending the acknowledge byte, the BSL enters a loop to receive 32 bytes via ASC0. These bytes are stored sequentially into locations 00'FA40_H through 00'FA5F_H of the IRAM, allowing up to 16 instructions to be placed into the RAM area. To execute the loaded code the BSL then jumps to location 00'FA40_H, that is, the first loaded instruction. The bootstrap loading sequence is now terminated; however, the ST10F276 remains in BSL mode. The initially loaded routine will most probably load additional code or data, as an average application is likely to require substantially more than 16 instructions. This second receive loop may directly use the pre-initialized interface ASC0 to receive data and store it in arbitrary user-defined locations.

This second level of loaded code may be

- the final application code
- another, more sophisticated, loader routine that adds a transmission protocol to enhance the integrity of the loaded code or data
- a code sequence to change the system configuration and enable the bus interface to store the received data into external memory

This process may go through several iterations or may directly execute the final application. In all cases the ST10F276 still runs in BSL mode, that is, with the watchdog timer disabled and limited access to the internal Flash area. All code fetches from the internal IFLASH area (01'0000_H...08'FFFF_H) are redirected to the special Test-Flash. Data read operations access the internal Flash of the ST10F276.

5.3.5 Choosing the baud rate for the BSL via UART

The calculation of the serial baud rate for ASC0 from the length of the first zero byte that is received allows the operation of the bootstrap loader of the ST10F276 with a wide range of baud rates. However, the upper and lower limits must be kept to ensure proper data transfer.

$$B_{\text{ST10F276}} = \frac{f_{\text{CPU}}}{32 \cdot (S0BRL + 1)}$$

The ST10F276 uses timer T6 to measure the length of the initial zero byte. The quantization uncertainty of this measurement implies the first deviation from the real baud rate; the next deviation is implied by the computation of the S0BRL reload value from the timer contents. The formula below shows the association:

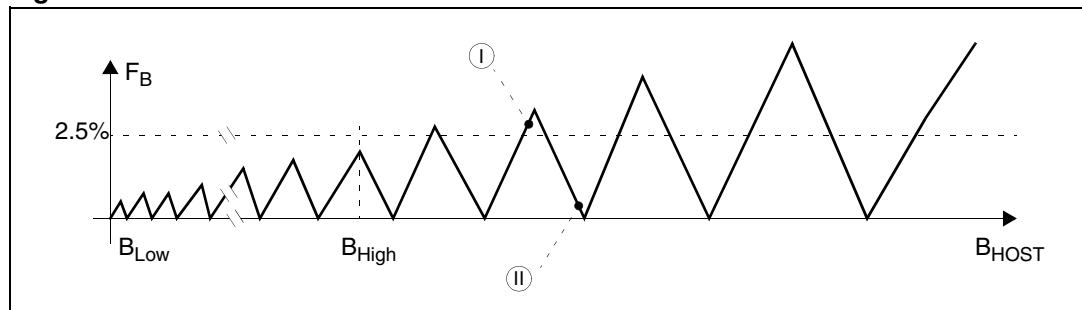
$$S0BRL = \frac{T6 - 36}{72}, \quad T6 = \frac{9}{4} \cdot \frac{f_{\text{CPU}}}{B_{\text{Host}}}$$

For a correct data transfer from the host to the ST10F276, the maximum deviation between the internal initialized baud rate for ASC0 and the real baud rate of the host should be below 2.5%. The deviation (F_B , in percent) between host baud rate and ST10F276 baud rate can be calculated using the formula below:

Note: Function (F_B) does not consider the tolerances of oscillators and other devices supporting the serial communication.

This baud rate deviation is a nonlinear function depending on the CPU clock and the baud rate of the host. The maxima of the function (F_B) increases with the host baud rate due to the smaller baud rate prescaler factors and the implied higher quantization error (see Figure 10).

Figure 10. Baud rate deviation between host and ST10F276



The minimum baud rate (B_{Low} in Figure 10) is determined by the maximum count capacity of timer T6, when measuring the zero byte, that is, it depends on the CPU clock. Using the maximum T6 count 2^{16} in the formula the minimum baud rate is calculated. The lowest

standard baud rate in this case would be 1200 baud. Baud rates below B_{Low} would cause T6 to overflow. In this case, ASC0 cannot be initialized properly.

The maximum baud rate (B_{High} in [Figure 10](#)) is the highest baud rate where the deviation still does not exceed the limit, that is, all baud rates between B_{Low} and B_{High} are below the deviation limit. The maximum standard baud rate that fulfills this requirement is 19200 baud.

Higher baud rates, however, may be used as long as the actual deviation does not exceed the limit. A certain baud rate (marked "I" in [Figure 10](#)) may, for example, violate the deviation limit, while an even higher baud rate (marked "II" in [Figure 10](#)) stays well below it. This depends on the host interface.

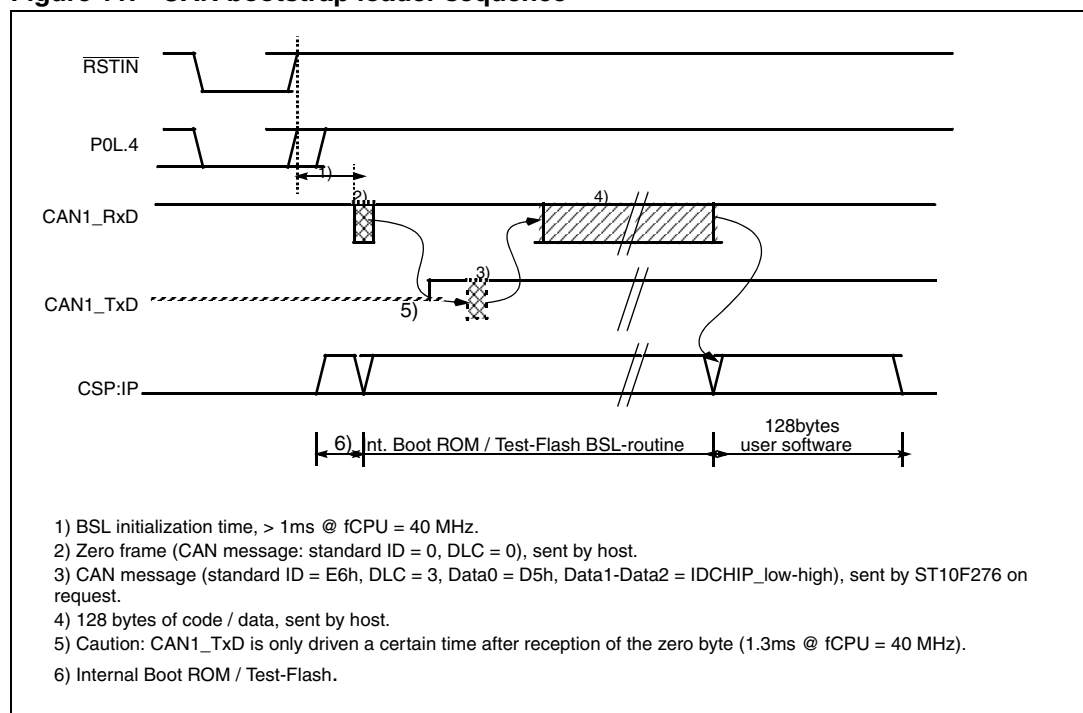
5.4 Standard bootstrap with CAN

5.4.1 Features

The bootstrap via CAN has the same overall behavior as the bootstrap via UART:

- Same bootstrapping steps;
- Same bootstrap method: Analyze the timing of a predefined frame, send back an acknowledge frame BUT only on request, load a fixed number of bytes and run;
- Same functionalities: Boot with different crystals and PLL ratios.

Figure 11. CAN bootstrap loader sequence



The Bootstrap Loader can load

- the complete application software into ROM-less systems,
- temporary software into complete systems for testing or calibration,
- a programming routine for Flash devices.

The BSL mechanism may be used for standard system start-up as well as for only special occasions like system maintenance (firmware update) or end-of-line programming or testing.

5.4.2 Entering the CAN bootstrap loader

The ST10F276 enters BSL mode if pin P0L.4 is sampled low at the end of a hardware reset. In this case, the built-in bootstrap loader is activated independently of the selected bus mode. The bootstrap loader code is stored in a special Test-Flash; no part of the standard mask ROM or Flash memory area is required for this.

After entering BSL mode and the respective initialization, the ST10F276 scans the CAN1_TxD line to receive the following initialization frame:

- Standard identifier = 0h
- DLC = 0h

As all the bits to be transmitted are dominant bits, a succession of 5 dominant bits and 1 stuff bit on the CAN network is used. From the duration of this frame, it calculates the corresponding baud rate factor with respect to the current CPU clock, initializes the CAN1 interface accordingly, switches pin CAN1_TxD to output and enables the CAN1 interface to take part in the network communication. Using this baud rate, a Message Object is configured in order to send an acknowledge frame. The ST10F276 will not send this Message Object but the host can request it by sending a remote frame.

The acknowledge frame is the following for the ST10F276:

- Standard identifier = E6h
- DLC = 3h
- Data0 = D5h, that is, generic acknowledge of the ST10 devices
- Data1 = IDCHIP least significant byte
- Data2 = IDCHIP most significant byte

For the ST10F276, IDCHIP = 114Xh.

Note: *Two behaviors can be distinguished in ST10 acknowledging to the host. If the host is behaving according to the CAN protocol, as at the beginning the ST10 CAN is not configured, the host is alone on the CAN network and does not receive an acknowledge. It automatically resends the zero frame. As soon as the ST10 CAN is configured, it acknowledges the zero frame. The “acknowledge frame” with identifier 0xE6 is configured, but the Transmit Request is not set. The host can request this frame to be sent and therefore obtains the IDCHIP by sending a remote frame.*

Hint: As the IDCHIP is sent in the acknowledge frame, Flash programming software now can immediately identify the exact type of device to be programmed.

5.4.3 ST10 configuration in CAN BSL

When the ST10F276 enters BSL mode via CAN, the configuration shown in [Table 31](#) is automatically set (values that deviate from the normal reset values are marked in **bold**).

Table 31. ST10 configuration in CAN BSL

| Function or register | Access | Notes |
|------------------------------|--|---|
| Watchdog timer | Disabled | |
| Register SYSCON | 0404_H ⁽¹⁾ | XPEN bit set |
| Context pointer CP | FA00 _H | |
| Register STKUN | FA00 _H | |
| Stack pointer SP | FA40 _H | |
| Register STKOV | FC00 _H | |
| Register BUSCON0 | acc. to startup config. ⁽²⁾ | |
| CAN1 Status/Control register | 0000 _H | Initialized only if Bootstrap via CAN |
| CAN1 Bit timing register | acc. to '0' frame | Initialized only if Bootstrap via CAN |
| XPERCON | 042D _H | XRAM1-2, XFlash, CAN1 and XMISC enabled |
| P4.6 / CAN1_TxD | '1' | Initialized only if Bootstrap via CAN |
| DP4.6 | '1' | Initialized only if Bootstrap via CAN |

1. In Bootstrap modes (standard or alternate) ROMEN, bit 10 of SYSCON, is always set regardless of \overline{EA} pin level. BYTDIS, bit 9 of SYSCON, is set according to data bus width selection via Port0 configuration.
2. BUSCON0 is initialized with 0000h, external bus disabled, if pin \overline{EA} is high during reset. If pin \overline{EA} is low during reset, BUSACT0, bit 10, and ALECTL0, bit 9, are set enabling the external bus with lengthened ALE signal. BTYP field, bit 7 and 6, is set according to Port0 configuration.

Other than after a normal reset, the watchdog timer is disabled, so the bootstrap loading sequence is not time limited. Pin CAN1_TxD1 is configured as output, so the ST10F276 can return the identification frame. Even if the internal IFLASH is enabled, a code cannot be executed from it.

5.4.4 Loading the start-up code via CAN

After sending the acknowledge byte, the BSL enters a loop to receive 128 bytes via CAN1.

Hint: The number of bytes loaded when booting via the CAN interface has been extended to 128 bytes to allow the reconfiguration of the CAN Bit Timing Register with the best timings (synchronization window, ...). This can be achieved by the following sequence of instructions:

ReconfigureBaud rate:

```
MOV R1, #041h
MOV DPP3:0EF00h, R1; Put CAN in Init, enable Configuration Change
MOV R1, #01600h
MOV DPP3:0EF06h, R1; 1MBaud at Fcpu = 20 MHz
```

These 128 bytes are stored sequentially into locations 00'FA40_H through 00'FABF_H of the IRAM, allowing up to 64 instructions to be placed into the RAM area. To execute the loaded code the BSL then jumps to location 00'FA40_H, that is, the first loaded instruction. The bootstrap loading sequence is now terminated; however, the ST10F276 remains in BSL

mode. Most probably the initially loaded routine will load additional code or data, as an average application is likely to require substantially more than 64 instructions. This second receive loop may directly use the pre-initialized CAN interface to receive data and store it in arbitrary user-defined locations.

This second level of loaded code may be

- the final application code
- another, more sophisticated, loader routine that adds a transmission protocol to enhance the integrity of the loaded code or data
- a code sequence to change the system configuration and enable the bus interface to store the received data into external memory

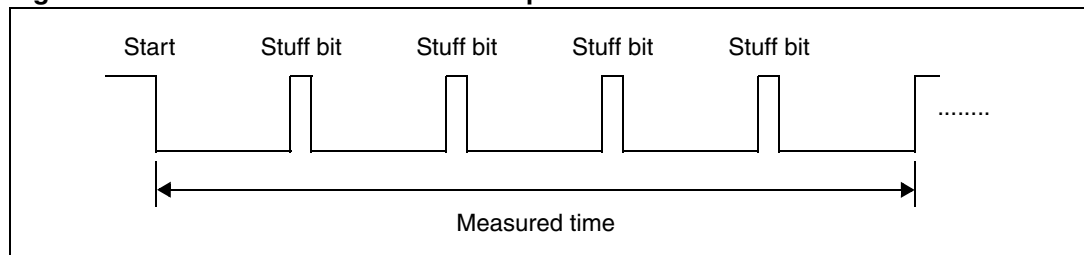
This process may go through several iterations or may directly execute the final application. In all cases the ST10F276 still runs in BSL mode, that is, with the watchdog timer disabled and limited access to the internal Flash area. All code fetches from the internal Flash area (01'0000_H ...08'FFFF_H) are redirected to the special Test-Flash. Data read operations will access the internal Flash of the ST10F276.

5.4.5 Choosing the baud rate for the BSL via CAN

The Bootstrap via CAN acts the same way as in the UART bootstrap mode. When the ST10F276 is started in BSL mode, it polls the RxD0 and CAN1_RxD lines. When polling a low level on one of these lines, a timer is launched that is stopped when the line returns to high level.

For CAN communication, the algorithm is made to receive a zero frame, that is, the standard identifier is 0x0, DLC is 0. This frame produces the following levels on the network: 5D, 1R, 5D, 1R, 5D, 1R, 5D, 1R, 5D, 1R, 4D, 1R, 1D, 11R. The algorithm lets the timer run until the detection of the 5th recessive bit. This way the bit timing is calculated over the duration of 29 bit times: This minimizes the error introduced by the polling.

Figure 12. Bit rate measurement over a predefined zero-frame



Error induced by the polling

The code used for the polling is the following:

```

WaitCom:
    JNB  P4.5,CAN_Boot      ; if SOF detected on CAN, then go to
CAN                                           ; loader
                                           ; Wait for start bit at RxD0
    JB   P3.11,WaitCom      ; Start Timer T6
    BSET T6R
    ....
CAN_Boot:
    BSET PWMCON.0           ; Start PWM Timer0
                                           ; (resolution is 1 CPU clk cycle)
    JMPR cc_UC,WaitRecessiveBit
WaitDominantBit:
    JB   P4.5,WaitDominantBit; wait for end of stuff bit
WaitRecessiveBit:
    JNB  P4.5,WaitRecessiveBit; wait for 1st dominant bit = Stuff
bit
    CMPI1R1,#5              ; Test if 5th stuff bit detected
    JMPR cc_NE,WaitDominantBit; No, go back to count more
    BCLR PWMCON.0           ; Stop timer
                                           ; here the 5th stuff bit is detected:
                                           ; PT0 = 29 Bit_Time (25D and 4R)

```

Therefore the maximum error at the detection of the communication on CAN pin is:

(1 not taken + 1 taken jumps) + 1 taken jump + 1 bit set: (6) + 6 CPU clock cycles

The error at the detection for the 5th recessive bit is:

(1 taken jump) + 1 not taken jump + 1 compare + 1 bit clear: (4) + 6 CPU cycles

In the worst case, the induced error is 6 CPU clock cycles, so the polling could induce an error of 6 timer ticks.

Error induced by the baud rate calculation

The content of the timer PT0 counter corresponds to 29 bit times, resulting in the following equation:

$$PT0 = 58 \times (BRP + 1) \times (1 + Tseg1 + Tseg2)$$

where BRP, Tseg1 and Tseg2 are the field of the CAN Bit Timing register.

The CAN protocol specification recommends to implement a bit time composed of at least 8 time quanta (tq). This recommendation is applied here. Moreover, the maximum bit time length is 25 tq. To ensure precision, the aim is to have the smallest Bit Rate Prescaler (BRP) and the maximum number of tq in a bit time.

This gives the following ranges for PT0 according to BRP:

$$8 \leq 1 + Tseg1 + Tseg2 \leq 25$$

$$464 \times (1 + BRP) \leq PT0 \leq 1450 \times (1 + BRP)$$

Table 32. BRP and PT0 values

| BRP | PT0_min | PT0_max | Comments |
|-----|---------|---------|-------------------------|
| 0 | 464 | 1450 | |
| 1 | 1451 | 2900 | |
| 2 | 2901 | 4350 | |
| 3 | 4351 | 5800 | |
| 4 | 5801 | 7250 | |
| 5 | 7251 | 8700 | |
| .. | .. | .. | |
| 43 | 20416 | 63800 | |
| 44 | 20880 | 65250 | |
| 45 | 21344 | 66700 | Possible timer overflow |
| .. | .. | .. | |
| 63 | X | X | |

The error coming from the measurement of the 29 bit is:

$$e_1 = 6 / [PT0]$$

It is maximal for the smallest BRP value and the smallest number of ticks in PT0. Therefore:

$$e_{1 \text{ Max}} = 1.29\%$$

To improve precision, the aim is to have the smallest BRP so that the time quantum is the smallest possible. Thus, an error on the calculation of time quanta in a bit time is minimal.

In order to do so, the value of PT0 is divided into ranges of 1450 ticks. In the algorithm, PT0 is divided by 1451 and the result is BRP.

The calculated BRP value is then used to divide PT0 in order to have the value of $(1 + Tseg1 + Tseg2)$. A table is made to set the values for $Tseg1$ and $Tseg2$ according to the value of $(1 + Tseg1 + Tseg2)$. These values of $Tseg1$ and $Tseg2$ are chosen in order to reach a sample point between 70% and 80% of the bit time.

During the calculation of $(1 + Tseg1 + Tseg2)$, an error e_2 can be introduced by the division. This error is of 1 time quantum maximum.

To compensate for any possible error on bit rate, the (Re)Synchronization Jump Width is fixed to 2 time quanta.

5.4.6 Computing the baud rate error

Considering the following conditions, a computation of the error is given as an example.

- CPU frequency: 20 MHz
- Target Bit Rate: 1 Mbit/s

In these conditions, the content of PT0 timer for 29 bits should be:

$$[PT0] = \frac{29 \times F_{cpu}}{BitRate} = \frac{29 \times 20 \times 10^6}{1 \times 10^6} = 580$$

Therefore:

$$574 < [PT0] < 586$$

This gives:

$$BRP = 0$$

$$tq = 100 \text{ ns}$$

Computation of $1 + Tseg1 + Tseg2$: Considering the equation:

$$[PT0] = 58 \times (1 + BRP) \times (1 + Tseg1 + Tseg2)$$

Thus:

$$9 = \frac{574}{58} \leq Tseg1 + Tseg2 \leq \frac{586}{58} = 10$$

In the algorithm, a rounding up to the superior value is made if the remainder of the division is greater than half of the divisor. This would have been the case if the PT0 content was 574. Thus, in this example the result is $1 + Tseg1 + Tseg2 = 10$, giving a bit time of exactly 1 μ s => no error in bit rate.

Note: In most cases (24 MHz, 32 MHz, 40 MHz of CPU frequency and 125, 250, 500 or 1Mb/s of bit rate), there is no error. Nevertheless, it is better to check for an error with the real application parameters.

The content of the bit timing register is: 0x1640. This gives a sample point at 80%.

Note: The (Re)Synchronization Jump Width is fixed to 2 time quanta.

5.4.7 Bootstrap via CAN

After the bootstrap phase, the ST10F276 CAN module is configured as follows:

- The pin P4.6 is configured as output (the latch value is '1' = recessive) to assume CAN1_TxD function.
- The MO2 is configured to output the acknowledge of the bootstrap with the standard identifier E6h, a DLC of 3 and Data0 = D5h, Data1 and 2 = IDCHIP.
- The MO1 is configured to receive messages with the standard identifier 5h. Its acceptance mask is set to ensure that all bits match. The DLC received is not checked: The ST10 expects only 1 byte of data at a time.

No other message is sent by the ST10F276 after the acknowledge.

Note: The CAN boot waits for 128 bytes of data instead of 32 bytes (see UART boot). This is done to allow the User to reconfigure the CAN bit rate as soon as possible.

5.5 Comparing the old and the new bootstrap loader

The following tables summarizes the differences between the old ST10 (boot via UART only) bootstrap and the new one (boot via UART or CAN).

Table 33. Software topics summary

| Old bootstrap loader | New bootstrap loader | Comments |
|---|---|--|
| Uses only 32 bytes in Dual-Port RAM from 00'FA40h | Uses up to 128 bytes in Dual-Port RAM from 00'FA40h | For compatibility between boot via UART and boot via CAN1, please avoid loading the application software in the 00'FA60h/00'FABFh range. |
| Loads 32 bytes from UART | Loads 32 bytes from UART (boot via UART mode) | Same files can be used for boot via UART. |
| User selected Xperipherals can be enabled during boot (Step 3 or Step 4). | Xperipherals selection is fixed. | User can change the Xperipherals selections through a specific code. |

5.5.1 Software aspects

As the CAN1 is needed, XPERCON register is configured by the bootstrap loader code and bit XPEN of SYSCON is set. However, as long as the EINIT instruction is not executed (and it is not in the bootstrap loader code), the settings can be modified. To do this, perform the following steps:

- Disable the XPeripherals by clearing XPEN in SYSCON register. Attention: If this part of the code is located in XRAM, it will be disabled.
- Enable the needed XPeripherals by writing the correct value in XPERCON register.
- Set XPEN bit in SYSCON.

5.5.2 Hardware aspects

Although the new bootstrap loader is designed to be compatible with the old bootstrap loader, there are a few hardware requirements for the new bootstrap loader as summarized in [Table 34](#).

Table 34. Hardware topics summary

| Actual bootstrap loader | New bootstrap loader | Comments |
|--|--|----------------------------------|
| P4.5 can be used as output in BSL mode. | P4.5 cannot be used as user output in BSL mode, but only as CAN1_RxD or input or address segments. | |
| Level on CAN1_RxD can change during boot Step 2. | Level on CAN1_RxD must be stable at '1' during boot Step 2. | External pull-up on P4.5 needed. |

5.6 Alternate boot mode (ABM)

5.6.1 Activation

Alternate boot is activated with the combination '01' on Port0L[5..4] at the rising edge of RSTIN.

5.6.2 Memory mapping

The ST10F276 has the same memory mapping for standard boot mode and for alternate boot mode:

- Test-Flash: Mapped from 00'0000h. The Standard Bootstrap Loader can be started by executing a jump to the address of this routine (JMPS 00'xxxx; *address to be defined*).
- User Flash: The User Flash is divided in two parts: The IFLASH, visible only for memory reads and memory writes (no code fetch) and the XFLASH, visible for any ST10 access (memory read, memory write and code fetch).
- All ST10F276 XRAM and Xperipherals modules can be accessed if enabled in XPERCON register.

Note: The alternate boot mode can be used to reprogram the whole content of the ST10F276 User Flash (except Block 0 in Bank 2, where the alternate boot is mapped into).

5.6.3 Interrupts

The ST10 interrupt vector table is always mapped from address 00'0000h.

As a consequence, interrupts are not allowed in Alternate Boot Mode; all maskable and nonmaskable interrupts must be disabled.

5.6.4 ST10 configuration in alternate boot mode

When the ST10F276 enters BSL mode via CAN, the configuration shown in [Table 35](#) is automatically set (values that deviate from the normal reset values are marked in **bold**).

Table 35. ST10 configuration in alternate boot mode

| Function or register | Access | Notes |
|----------------------|--|-------------------------------|
| Watchdog timer | Disabled | |
| Register SYSCON | 0404H ⁽¹⁾ | XPEN bit set |
| Context pointer CP | FA00 _H | |
| Register STKUN | FA00 _H | |
| Stack pointer SP | FA40 _H | |
| Register STKOV | FC00 _H | |
| Register BUSCON0 | acc. to startup config. ⁽²⁾ | |
| XPERCON | 002D _H | XRAM1-2, XFlash, CAN1 enabled |

1. In Bootstrap modes (standard or alternate) ROMEN, bit 10 of SYSCON, is always set regardless of \overline{EA} pin level. BYTDIS, bit 9 of SYSCON, is set according to data bus width selection via Port0 configuration.
2. BUSCON0 is initialized with 0000h, external bus disabled, if pin \overline{EA} is high during reset. If pin \overline{EA} is low during reset, BUSACT0, bit 10, and ALECTL0, bit 9, are set enabling the external bus with lengthened ALE signal. BTYP field, bit 7 and 6, is set according to Port0 configuration.

Even if the internal IFLASH is enabled, a code cannot be executed from it.

As the XFlash is needed, XPERCON register is configured by the ABM loader code and bit XPEN of SYSCON is set. However, as long as the EINIT instruction is not executed (and it is not in the bootstrap loader code), the settings can be modified. To do this, perform the following steps:

- Copy in DPRAM a function that will
 - disable the XPeripherals by clearing XPEN in SYSCON register,
 - enabled the needed XPeripherals by writing the correct value in XPERCON register,
 - set XPEN bit in SYSCON,
 - return to calling address.
- Call the function from XFlash

The changing of the XPERCON value cannot be executed from the XFlash because the XFlash is disabled by the clearing of XPEN bit in SYSCON.

5.6.5 Watchdog

As for standard boot, the Watchdog timer remains disabled during Alternate Boot Mode. In case a Watchdog reset occurs, a software reset is generated.

Note: See note from [Section 5.2.7](#) concerning software reset.

5.6.6 Exiting alternate boot mode

Once the ABM mode is entered, it can be exited only with a software or hardware reset.

Note: See note from [Section 5.2.7](#) concerning software reset.

5.6.7 Alternate boot user software

If the rules described previously are respected (that is, mapping of variables, disabling of interrupts, exit conditions, predefined vectors in Block 0 of Bank 2, Watchdog usage), then users can write the software they want to execute in this mode starting from 09'0000h.

5.6.8 User/alternate mode signature integrity check

The behavior of the Alternate Boot Mode is based on the computing of a signature between the content of two memory locations and a comparison with a reference signature. This requires that users who use Alternate Boot have reserved and programmed the Flash memory locations according to:

User mode signature

00'0000h: memory address of *operand0* for the signature computing

00'1FFCh: memory address of *operand1* for the signature computing

00'1FFEh: memory address for the signature reference

Alternate mode signature

09'0000h: memory address of *operand0* for the signature computing

09'1FFCh: memory address of *operand1* for the signature computing

09'1FFEh: memory address for the signature reference

The values for *operand0*, *operand1* and the signature should be such that the sequence shown in the figure below is successfully executed.

```

MOV    Rx, CheckBlock1Addr; 00'0000h for standard reset
ADD    Rx, CheckBlock2Addr; 00'1FFCh for standard reset
CPLB   RLx                ; 1s complement of the lower
                           ; byte of the sum
CMP     Rx, CheckBlock3Addr; 00'1FFEh for standard reset

```

5.6.9 Alternate boot user software aspects

User defined alternate boot code must start at 09'0000h. A new SFR created on the ST10F276 indicates that the device is running in Alternate Boot Mode: Bit 5 of EMUCON (mapped at 0xFE0Ah) is set when the alternate boot is selected by the reset configuration. All the other bits are ignored when checking the content of this register to read the value of bit 5.

This bit is a read-only bit. It remains set until the next software or hardware reset.

5.6.10 EMUCON register

| EMUCON (FE0Ah / 05h) | | | | | | | | SFR | | | | Reset value: - xxh: | | | |
|----------------------|----|----|----|----|----|---|---|-----|---|-----|---|---------------------|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | | | | | | | | | | ABM | | | | | |
| | | | | | | | | | | R | | | | | |

Table 36. ABM bit description

| Bit | Function |
|-----|---|
| ABM | ABM Flag (or TMOD3) '0': Alternate Boot Mode is not selected by reset configuration on P0L[5..4] '1': Alternate Boot Mode is selected by reset configuration on P0L[5..4]: This bit is set if P0L[5..4] = '01' during hardware reset. |

5.6.11 Internal decoding of test modes

The test mode decoding logic is located inside the ST10F276 Bus Controller.

The decoding is as follows:

- Alternate Boot Mode decoding: ($\overline{P0L.5}$ & P0L.4)
- Standard Bootstrap decoding: (P0L.5 & $\overline{P0L.4}$)
- Normal operation: (P0L.5 & P0L.4)

The other configurations select ST internal test modes.

5.6.12 Example

In the following example, Alternate Boot Mode works as follows:

- On rising edge of \overline{RSTIN} pin, the reset configuration is latched.
- If Bootstrap Loader mode is not enabled (P0L[5..4] = '11'), ST10F276 hardware proceeds with a standard hardware reset procedure.
- If standard Bootstrap Loader is enabled (P0L[5..4] = '10'), the standard ST10 Bootstrap Loader is enabled and a variable is cleared to indicate that ABM is not enabled.
- If Alternate Boot Mode is selected (P0L[5..4] = '01'), then, depending on signatures integrity checks, a predefined reset sequence is activated.

5.7 Selective boot mode

The selective boot is a subcase of the Alternate Boot Mode. When none of the signatures are correct, instead of executing the standard bootstrap loader (triggered by P0L.4 low at reset), an additional check is made.

Address 00'1FFCh is read again with the following behavior:

- If value is 0000h or FFFFh, a jump is performed to the standard bootstrap loader.
- Otherwise:
 - High byte is disregarded.
 - Low byte bits select which communication channel is enabled.

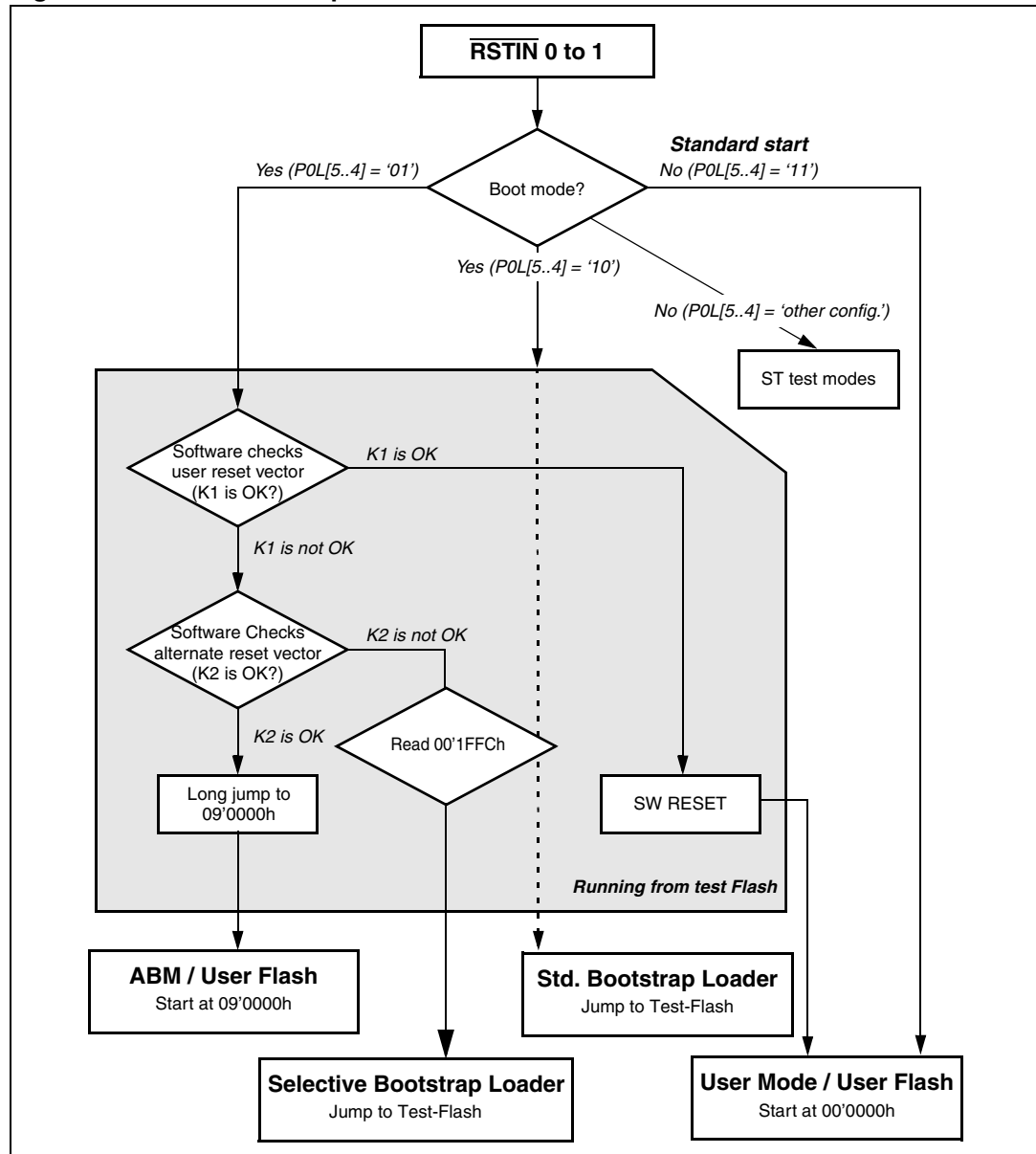
Table 37. Selective boot

| Bit | Function |
|------|--|
| 0 | UART selection '0': UART is not watched for a Start condition. '1': UART is watched for a Start condition. |
| 1 | CAN1 selection '0': CAN1 is not watched for a Start condition. '1': CAN1 is watched for a Start condition. |
| 2..7 | Reserved For upward compatibility, must be programmed to '0' |

Therefore a value:

- 0xXX03 configures the Selective Bootstrap Loader to poll for RxD0 and CAN1_RxD.
- 0xXX01 configures the Selective Bootstrap Loader to poll only RxD0 (no boot via CAN).
- 0xXX02 configures the Selective Bootstrap Loader to poll only CAN1_RxD (no boot via UART).
- Other values allow the ST10F276 to execute an endless loop into the Test-Flash.

Figure 13. Reset boot sequence



6 Central processing unit (CPU)

The CPU includes a 4-stage instruction pipeline, a 16-bit arithmetic and logic unit (ALU) and dedicated SFRs. Additional hardware has been added for a separate multiply and divide unit, a bit-mask generator and a barrel shifter.

Most of the ST10F276's instructions can be executed in one instruction cycle which requires 31.25ns at 64 MHz CPU clock. For example, shift and rotate instructions are processed in one instruction cycle independent of the number of bits to be shifted.

Multiple-cycle instructions have been optimized: branches are carried out in 2 cycles, 16 x 16-bit multiplication in 5 cycles and a 32/16-bit division in 10 cycles.

The jump cache reduces the execution time of repeatedly performed jumps in a loop, from 2 cycles to 1 cycle.

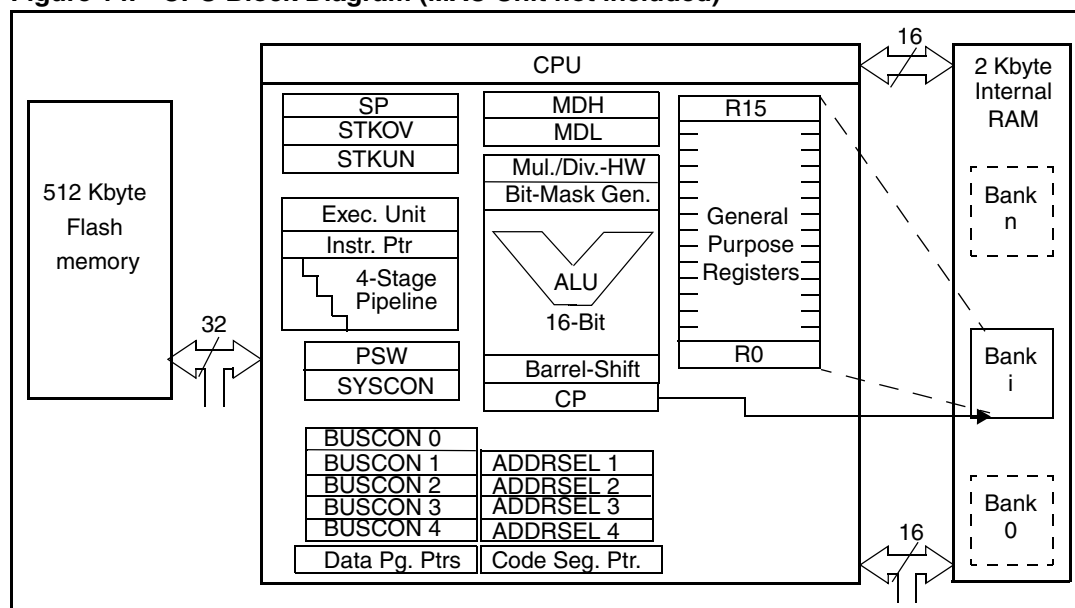
The CPU uses a bank of 16 word registers to run the current context. This bank of General Purpose Registers (GPR) is physically stored within the on-chip Internal RAM (IRAM) area. A Context Pointer (CP) register determines the base address of the active register bank to be accessed by the CPU.

The number of register banks is only restricted by the available Internal RAM space. For easy parameter passing, a register bank may overlap others.

A system stack of up to 2048 bytes is provided as a storage for temporary data. The system stack is allocated in the on-chip RAM area, and it is accessed by the CPU via the stack pointer (SP) register.

Two separate SFRs, STKOV and STKUN, are implicitly compared against the stack pointer value upon each stack access for the detection of a stack overflow or underflow.

Figure 14. CPU Block Diagram (MAC Unit not included)



6.1 Multiplier-accumulator unit (MAC)

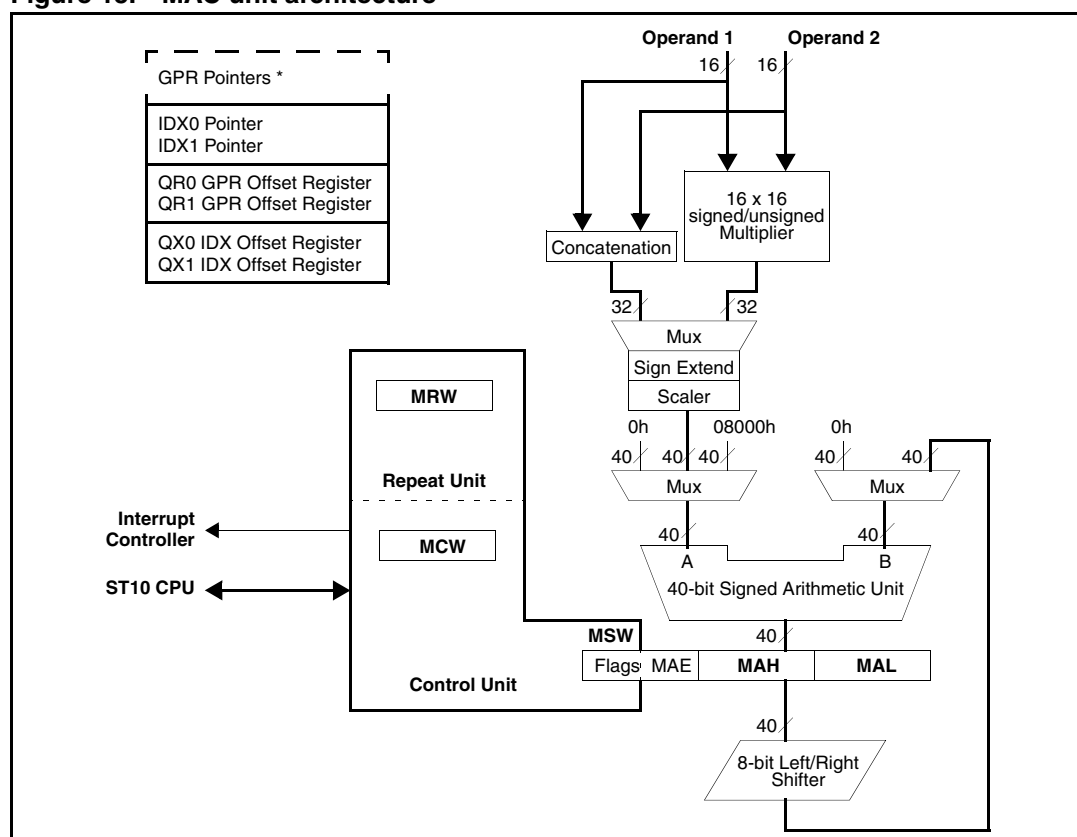
The MAC coprocessor is a specialized coprocessor added to the ST10 CPU Core in order to improve the performances of the ST10 Family in signal processing algorithms.

The standard ST10 CPU has been modified to include new addressing capabilities which enable the CPU to supply the new coprocessor with up to 2 operands per instruction cycle.

This new coprocessor (so-called MAC) contains a fast multiply-accumulate unit and a repeat unit.

The coprocessor instructions extend the ST10 CPU instruction set with multiply, multiply-accumulate, 32-bit signed arithmetic operations.

Figure 15. MAC unit architecture



6.2 Instruction set summary

The [Table 38](#) lists the instructions of the ST10F276. The detailed description of each instruction can be found in the “ST10 Family Programming Manual”.

Table 38. Standard instruction set summary

| Mnemonic | Description | Bytes |
|------------------|---|-------|
| ADD(B) | Add word (byte) operands | 2 / 4 |
| ADDC(B) | Add word (byte) operands with Carry | 2 / 4 |
| SUB(B) | Subtract word (byte) operands | 2 / 4 |
| SUBC(B) | Subtract word (byte) operands with Carry | 2 / 4 |
| MUL(U) | (Un)Signed multiply direct GPR by direct GPR (16-/16-bit) | 2 |
| DIV(U) | (Un)Signed divide register MDL by direct GPR (16-/16-bit) | 2 |
| DIVL(U) | (Un)Signed long divide reg. MD by direct GPR (32-/16-bit) | 2 |
| CPL(B) | Complement direct word (byte) GPR | 2 |
| NEG(B) | Negate direct word (byte) GPR | 2 |
| AND(B) | Bit-wise AND, (word/byte operands) | 2 / 4 |
| OR(B) | Bit-wise OR, (word/byte operands) | 2 / 4 |
| XOR(B) | Bit-wise XOR, (word/byte operands) | 2 / 4 |
| BCLR | Clear direct bit | 2 |
| BSET | Set direct bit | 2 |
| BMOV(N) | Move (negated) direct bit to direct bit | 4 |
| BAND, BOR, BXOR | AND/OR/XOR direct bit with direct bit | 4 |
| BCMP | Compare direct bit to direct bit | 4 |
| BFLDH/L | Bit-wise modify masked high/low byte of bit-addressable direct word memory with immediate data | 4 |
| CMP(B) | Compare word (byte) operands | 2 / 4 |
| CMPD1/2 | Compare word data to GPR and decrement GPR by 1/2 | 2 / 4 |
| CMPI1/2 | Compare word data to GPR and increment GPR by 1/2 | 2 / 4 |
| PRIOR | Determine number of shift cycles to normalize direct word GPR and store result in direct word GPR | 2 |
| SHL / SHR | Shift left/right direct word GPR | 2 |
| ROL / ROR | Rotate left/right direct word GPR | 2 |
| ASHR | Arithmetic (sign bit) shift right direct word GPR | 2 |
| MOV(B) | Move word (byte) data | 2 / 4 |
| MOVBS | Move byte operand to word operand with sign extension | 2 / 4 |
| MOVBZ | Move byte operand to word operand with zero extension | 2 / 4 |
| JMPA, JMPI, JMPR | Jump absolute/indirect/relative if condition is met | 4 |
| JMPS | Jump absolute to a code segment | 4 |

Table 38. Standard instruction set summary (continued)

| Mnemonic | Description | Bytes |
|---------------------|---|-------|
| J(N)B | Jump relative if direct bit is (not) set | 4 |
| JBC | Jump relative and clear bit if direct bit is set | 4 |
| JNBS | Jump relative and set bit if direct bit is not set | 4 |
| CALLA, CALLI, CALLR | Call absolute/indirect/relative subroutine if condition is met | 4 |
| CALLS | Call absolute subroutine in any code segment | 4 |
| PCALL | Push direct word register onto system stack and call absolute subroutine | 4 |
| TRAP | Call interrupt service routine via immediate trap number | 2 |
| PUSH, POP | Push/pop direct word register onto/from system stack | 2 |
| SCXT | Push direct word register onto system stack and update register with word operand | 4 |
| RET | Return from intra-segment subroutine | 2 |
| RETS | Return from inter-segment subroutine | 2 |
| RETP | Return from intra-segment subroutine and pop direct word register from system stack | 2 |
| RETI | Return from interrupt service subroutine | 2 |
| SRST | Software Reset | 4 |
| IDLE | Enter Idle Mode | 4 |
| PWRDN | Enter Power Down Mode (supposes $\overline{\text{NMI}}$ -pin being low) | 4 |
| SRVWDT | Service Watchdog Timer | 4 |
| DISWDT | Disable Watchdog Timer | 4 |
| EINIT | Signify End-of-Initialization on RSTOUT-pin | 4 |
| ATOMIC | Begin ATOMIC sequence | 2 |
| EXTR | Begin EXTended Register sequence | 2 |
| EXTP(R) | Begin EXTended Page (and Register) sequence | 2 / 4 |
| EXTS(R) | Begin EXTended Segment (and Register) sequence | 2 / 4 |
| NOP | Null operation | 2 |

6.3 MAC coprocessor specific instructions

The [Table 39](#) lists the MAC instructions of the ST10F276. The detailed description of each instruction can be found in the “ST10 Family Programming Manual”. Note that all MAC instructions are encoded on 4 bytes.

Table 39. MAC instruction set summary

| Mnemonic | Description |
|----------------------|--|
| CoABS | Absolute Value of the Accumulator |
| CoADD(2) | Addition |
| CoASHR(rnd) | Accumulator Arithmetic Shift Right & Optional Round |
| CoCMP | Compare Accumulator with Operands |
| CoLOAD(-,2) | Load Accumulator with Operands |
| CoMAC(R,u,s,-,rnd) | (Un)Signed/(Un)Signed Multiply-Accumulate & Optional Round |
| CoMACM(R)(u,s,-,rnd) | (Un)Signed/(Un)Signed Multiply-Accumulate with Parallel Data Move & Optional Round |
| CoMAX / CoMIN | Maximum / Minimum of Operands and Accumulator |
| CoMOV | Memory to Memory Move |
| CoMUL(u,s,-,rnd) | (Un)Signed/(Un)Signed multiply & Optional Round |
| CoNEG(rnd) | Negate Accumulator & Optional Round |
| CoNOP | No-Operation |
| CoRND | Round Accumulator |
| CoSHL / CoSHR | Accumulator Logical Shift Left / Right |
| CoSTORE | Store a MAC Unit Register |
| CoSUB(2,R) | Substraction |

7 External bus controller

All of the external memory accesses are performed by the on-chip external bus controller.

The EBC can be programmed to single chip mode when no external memory is required, or to one of four different external memory access modes:

- 16- / 18- / 20- / 24-bit addresses and 16-bit data, demultiplexed
- 16- / 18- / 20- / 24-bit addresses and 16-bit data, multiplexed
- 16- / 18- / 20- / 24-bit addresses and 8-bit data, multiplexed
- 16- / 18- / 20- / 24-bit addresses and 8-bit data, demultiplexed

In demultiplexed bus modes addresses are output on PORT1 and data is input / output on PORT0 or P0L, respectively. In the multiplexed bus modes both addresses and data use PORT0 for input / output.

Timing characteristics of the external bus interface (memory cycle time, memory tri-state time, length of ALE and read / write delay) are programmable giving the choice of a wide range of memories and external peripherals.

Up to four independent address windows may be defined (using register pairs ADDRSELx / BUSCONx) to access different resources and bus characteristics.

These address windows are arranged hierarchically where BUSCON4 overrides BUSCON3 and BUSCON2 overrides BUSCON1.

All accesses to locations not covered by these four address windows are controlled by BUSCON0. Up to five external \overline{CS} signals (four windows plus default) can be generated in order to save external glue logic. Access to very slow memories is supported by a 'Ready' function.

A \overline{HOLD} / \overline{HLDA} protocol is available for bus arbitration which shares external resources with other bus masters.

The bus arbitration is enabled by setting bit HLDEN in register PSW. After setting HLDEN once, pins P6.7...P6.5 (\overline{BREQ} , \overline{HLDA} , \overline{HOLD}) are automatically controlled by the EBC. In master mode (default after reset) the \overline{HLDA} pin is an output. By setting bit DP6.7 to '1' the slave mode is selected where pin \overline{HLDA} is switched to input. This directly connects the slave controller to another master controller without glue logic.

For applications which require less external memory space, the address space can be restricted to 1 Mbyte, 256 Kbytes or to 64 Kbytes. Port 4 outputs all eight address lines if an address space of 16M Bytes is used, otherwise four, two or no address lines.

Chip select timing can be made programmable. By default (after reset), the \overline{CSx} lines change half a CPU clock cycle after the rising edge of ALE. With the CSCFG bit set in the SYSCON register the \overline{CSx} lines change with the rising edge of ALE.

The active level of the READY pin can be set by bit RDYPOL in the BUSCONx registers. When the READY function is enabled for a specific address window, each bus cycle within the window must be terminated with the active level defined by bit RDYPOL in the associated BUSCON register.

8 Interrupt system

The interrupt response time for internal program execution is from 78ns to 187.5ns at 64 MHz CPU clock.

The ST10F276 architecture supports several mechanisms for fast and flexible response to service requests that can be generated from various sources (internal or external) to the microcontroller. Any of these interrupt requests can be serviced by the Interrupt Controller or by the Peripheral Event Controller (PEC).

In contrast to a standard interrupt service where the current program execution is suspended and a branch to the interrupt vector table is performed, just one cycle is 'stolen' from the current CPU activity to perform a PEC service. A PEC service implies a single Byte or Word data transfer between any two memory locations with an additional increment of either the PEC source or destination pointer. An individual PEC transfer counter is implicitly decremented for each PEC service except when performing in the continuous transfer mode. When this counter reaches zero, a standard interrupt is performed to the corresponding source related vector location. PEC services are very well suited to perform the transmission or the reception of blocks of data. The ST10F276 has 8 PEC channels, each of them offers such fast interrupt-driven data transfer capabilities.

An interrupt control register which contains an interrupt request flag, an interrupt enable flag and an interrupt priority bit-field is dedicated to each existing interrupt source. Thanks to its related register, each source can be programmed to one of sixteen interrupt priority levels. Once starting to be processed by the CPU, an interrupt service can only be interrupted by a higher prioritized service request. For the standard interrupt processing, each of the possible interrupt sources has a dedicated vector location.

Software interrupts are supported by means of the 'TRAP' instruction in combination with an individual trap (interrupt) number.

Fast external interrupt inputs are provided to service external interrupts with high precision requirements. These fast interrupt inputs feature programmable edge detection (rising edge, falling edge or both edges).

Fast external interrupts may also have interrupt sources selected from other peripherals; for example the CANx controller receive signals (CANx_RxD) and I²C serial clock signal can be used to interrupt the system.

Table 40 shows all the available ST10F276 interrupt sources and the corresponding hardware-related interrupt flags, vectors, vector locations and trap (interrupt) numbers:

Table 40. Interrupt sources

| Source of Interrupt or PEC Service Request | Request Flag | Enable Flag | Interrupt Vector | Vector Location | Trap Number |
|--|--------------|-------------|------------------|-----------------|-------------|
| CAPCOM Register 0 | CC0IR | CC0IE | CC0INT | 00'0040h | 10h |
| CAPCOM Register 1 | CC1IR | CC1IE | CC1INT | 00'0044h | 11h |
| CAPCOM Register 2 | CC2IR | CC2IE | CC2INT | 00'0048h | 12h |
| CAPCOM Register 3 | CC3IR | CC3IE | CC3INT | 00'004Ch | 13h |
| CAPCOM Register 4 | CC4IR | CC4IE | CC4INT | 00'0050h | 14h |
| CAPCOM Register 5 | CC5IR | CC5IE | CC5INT | 00'0054h | 15h |

Table 40. Interrupt sources (continued)

| Source of Interrupt or PEC Service Request | Request Flag | Enable Flag | Interrupt Vector | Vector Location | Trap Number |
|--|--------------|-------------|------------------|-----------------|-------------|
| CAPCOM Register 6 | CC6IR | CC6IE | CC6INT | 00'0058h | 16h |
| CAPCOM Register 7 | CC7IR | CC7IE | CC7INT | 00'005Ch | 17h |
| CAPCOM Register 8 | CC8IR | CC8IE | CC8INT | 00'0060h | 18h |
| CAPCOM Register 9 | CC9IR | CC9IE | CC9INT | 00'0064h | 19h |
| CAPCOM Register 10 | CC10IR | CC10IE | CC10INT | 00'0068h | 1Ah |
| CAPCOM Register 11 | CC11IR | CC11IE | CC11INT | 00'006Ch | 1Bh |
| CAPCOM Register 12 | CC12IR | CC12IE | CC12INT | 00'0070h | 1Ch |
| CAPCOM Register 13 | CC13IR | CC13IE | CC13INT | 00'0074h | 1Dh |
| CAPCOM Register 14 | CC14IR | CC14IE | CC14INT | 00'0078h | 1Eh |
| CAPCOM Register 15 | CC15IR | CC15IE | CC15INT | 00'007Ch | 1Fh |
| CAPCOM Register 16 | CC16IR | CC16IE | CC16INT | 00'00C0h | 30h |
| CAPCOM Register 17 | CC17IR | CC17IE | CC17INT | 00'00C4h | 31h |
| CAPCOM Register 18 | CC18IR | CC18IE | CC18INT | 00'00C8h | 32h |
| CAPCOM Register 19 | CC19IR | CC19IE | CC19INT | 00'00CCh | 33h |
| CAPCOM Register 20 | CC20IR | CC20IE | CC20INT | 00'00D0h | 34h |
| CAPCOM Register 21 | CC21IR | CC21IE | CC21INT | 00'00D4h | 35h |
| CAPCOM Register 22 | CC22IR | CC22IE | CC22INT | 00'00D8h | 36h |
| CAPCOM Register 23 | CC23IR | CC23IE | CC23INT | 00'00DCh | 37h |
| CAPCOM Register 24 | CC24IR | CC24IE | CC24INT | 00'00E0h | 38h |
| CAPCOM Register 25 | CC25IR | CC25IE | CC25INT | 00'00E4h | 39h |
| CAPCOM Register 26 | CC26IR | CC26IE | CC26INT | 00'00E8h | 3Ah |
| CAPCOM Register 27 | CC27IR | CC27IE | CC27INT | 00'00ECh | 3Bh |
| CAPCOM Register 28 | CC28IR | CC28IE | CC28INT | 00'00F0h | 3Ch |
| CAPCOM Register 29 | CC29IR | CC29IE | CC29INT | 00'0110h | 44h |
| CAPCOM Register 30 | CC30IR | CC30IE | CC30INT | 00'0114h | 45h |
| CAPCOM Register 31 | CC31IR | CC31IE | CC31INT | 00'0118h | 46h |
| CAPCOM Timer 0 | T0IR | T0IE | T0INT | 00'0080h | 20h |
| CAPCOM Timer 1 | T1IR | T1IE | T1INT | 00'0084h | 21h |
| CAPCOM Timer 7 | T7IR | T7IE | T7INT | 00'00F4h | 3Dh |
| CAPCOM Timer 8 | T8IR | T8IE | T8INT | 00'00F8h | 3Eh |
| GPT1 Timer 2 | T2IR | T2IE | T2INT | 00'0088h | 22h |
| GPT1 Timer 3 | T3IR | T3IE | T3INT | 00'008Ch | 23h |
| GPT1 Timer 4 | T4IR | T4IE | T4INT | 00'0090h | 24h |
| GPT2 Timer 5 | T5IR | T5IE | T5INT | 00'0094h | 25h |

Table 40. Interrupt sources (continued)

| Source of Interrupt or PEC Service Request | Request Flag | Enable Flag | Interrupt Vector | Vector Location | Trap Number |
|--|--------------|-------------|------------------|-----------------|-------------|
| GPT2 Timer 6 | T6IR | T6IE | T6INT | 00'0098h | 26h |
| GPT2 CAPREL Register | CRIR | CRIE | CRINT | 00'009Ch | 27h |
| A/D Conversion Complete | ADCIR | ADCIE | ADCINT | 00'00A0h | 28h |
| A/D Overrun Error | ADEIR | ADEIE | ADEINT | 00'00A4h | 29h |
| ASC0 Transmit | S0TIR | S0TIE | S0TINT | 00'00A8h | 2Ah |
| ASC0 Transmit Buffer | S0TBIR | S0TBIE | S0TBINT | 00'011Ch | 47h |
| ASC0 Receive | S0RIR | S0RIE | S0RINT | 00'00ACh | 2Bh |
| ASC0 Error | S0EIR | S0EIE | S0EINT | 00'00B0h | 2Ch |
| SSC Transmit | SCTIR | SCTIE | SCTINT | 00'00B4h | 2Dh |
| SSC Receive | SCRIR | SCRIE | SCRINT | 00'00B8h | 2Eh |
| SSC Error | SCEIR | SCEIE | SCEINT | 00'00BCh | 2Fh |
| PWM Channel 0...3 | PWMIR | PWMIE | PWMINT | 00'00FCh | 3Fh |
| See Paragraph 8.1 | XP0IR | XP0IE | XP0INT | 00'0100h | 40h |
| See Paragraph 8.1 | XP1IR | XP1IE | XP1INT | 00'0104h | 41h |
| See Paragraph 8.1 | XP2IR | XP2IE | XP2INT | 00'0108h | 42h |
| See Paragraph 8.1 | XP3IR | XP3IE | XP3INT | 00'010Ch | 43h |

Hardware traps are exceptions or error conditions that arise during run-time. They cause immediate non-maskable system reaction similar to a standard interrupt service (branching to a dedicated vector table location).

The occurrence of a hardware trap is additionally signified by an individual bit in the trap flag register (TFR). Except when another higher prioritized trap service is in progress, a hardware trap will interrupt any other program execution. Hardware trap services cannot not be interrupted by standard interrupt or by PEC interrupts.

8.1 X-Peripheral interrupt

The limited number of X-Bus interrupt lines of the present ST10 architecture, imposes some constraints on the implementation of the new functionality. In particular, the additional X-Peripherals SSC1, ASC1, I²C, PWM1 and RTC need some resources to implement interrupt and PEC transfer capabilities. For this reason, a multiplexed structure for the interrupt management is proposed. In the next [Figure 16](#), the principle is explained through a simple diagram, which shows the basic structure replicated for each of the four X-interrupt available vectors (XP0INT, XP1INT, XP2INT and XP3INT).

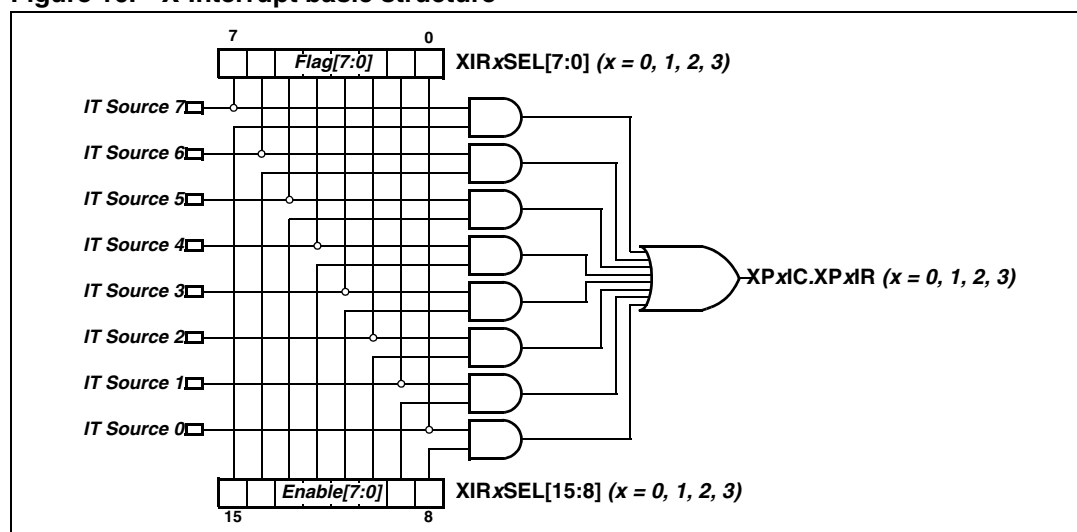
It is based on a set of 16-bit registers XIRxSEL (x=0,1,2,3), divided in two portions each:

- Byte High XIRxSEL[15:8] Interrupt Enable bits
- Byte Low XIRxSEL[7:0] Interrupt Flag bits

When different sources submit an interrupt request, the enable bits (Byte High of XIRxSEL register) define a mask which controls which sources will be associated with the unique

available vector. If more than one source is enabled to issue the request, the service routine will have to take care to identify the real event to be serviced. This can easily be done by checking the flag bits (Byte Low of XIRxSEL register). Note that the flag bits can also provide information about events which are not currently serviced by the interrupt controller (since masked through the enable bits), allowing an effective software management also in absence of the possibility to serve the related interrupt request: a periodic polling of the flag bits may be implemented inside the user application.

Figure 16. X-Interrupt basic structure



The [Table 41](#) summarizes the mapping of the different interrupt sources which shares the four X-interrupt vectors.

Table 41. X-Interrupt detailed mapping

| | XP0INT | XP1INT | XP2INT | XP3INT |
|----------------------|--------|--------|--------|--------|
| CAN1 Interrupt | x | | | x |
| CAN2 Interrupt | | x | | x |
| I2C Receive | x | x | x | |
| I2C Transmit | x | x | x | |
| I2C Error | | | | x |
| SSC1 Receive | x | x | x | |
| SSC1 Transmit | x | x | x | |
| SSC1 Error | | | | x |
| ASC1 Receive | x | x | x | |
| ASC1 Transmit | x | x | x | |
| ASC1 Transmit Buffer | x | x | x | |

Table 41. X-Interrupt detailed mapping (continued)

| | XP0INT | XP1INT | XP2INT | XP3INT |
|--------------------|--------|--------|--------|--------|
| ASC1 Error | | | | x |
| PLL Unlock / OWD | | | | x |
| PWM1 Channel 3...0 | | | x | x |

8.2 Exception and error traps list

[Table 42](#) shows all of the possible exceptions or error conditions that can arise during run-time.

Table 42. Trap priorities

| Exception Condition | Trap Flag | Trap Vector | Vector Location | Trap Number | Trap* Priority |
|--|--|--|--|--|----------------------------|
| Reset Functions: Hardware Reset Software Reset Watchdog Timer Overflow | | RESET RESET RESET | 00'0000h 00'0000h 00'0000h | 00h 00h 00h | III III III |
| Class A Hardware Traps: Non-Maskable Interrupt Stack Overflow Stack Underflow | NMI STKOF STKUF | NMITRAP STOTRAP STUTRAP | 00'0008h 00'0010h 00'0018h | 02h 04h 06h | II II II |
| Class B Hardware Traps: Undefined Opcode MAC Interruption Protected Instruction Fault Illegal word Operand Access Illegal Instruction Access Illegal External Bus Access | UNDOPC MACTRP PRTFLT ILLOPA ILLINA ILLBUS | BTRAP BTRAP BTRAP BTRAP BTRAP BTRAP | 00'0028h 00'0028h 00'0028h 00'0028h 00'0028h 00'0028h | 0Ah 0Ah 0Ah 0Ah 0Ah 0Ah | I I I I I I |
| Reserved | | | [002Ch - 003Ch] | [0Bh - 0Fh] | |
| Software Traps TRAP Instruction | | | Any 0000h – 01FCh in steps of 4h | Any [00h - 7Fh] | Current CPU Priority |

Note:

- * - All the class B traps have the same trap number (and vector) and the same lower priority compare to the class A traps and to the resets.
- Each class A traps has a dedicated trap number (and vector). They are prioritized in the second priority level.
- The resets have the highest priority level and the same trap number.
- The PSW.ILVL CPU priority is forced to the highest level (15) when these exceptions are serviced.

9 Capture / compare (CAPCOM) units

The ST10F276 has two 16-channel CAPCOM units which support generation and control of timing sequences on up to 32 channels with a maximum resolution of 125ns at 64 MHz CPU clock.

The CAPCOM units are typically used to handle high speed I/O tasks such as pulse and waveform generation, pulse width modulation (PMW), Digital to Analog (D/A) conversion, software timing, or time recording relative to external events.

Four 16-bit timers (T0/T1, T7/T8) with reload registers provide two independent time bases for the capture/compare register array.

The input clock for the timers is programmable to several prescaled values of the internal system clock, or may be derived from an overflow/underflow of timer T6 in module GPT2.

This provides a wide range of variation for the timer period and resolution and allows precise adjustments to application specific requirements. In addition, external count inputs for CAPCOM timers T0 and T7 allow event scheduling for the capture/compare registers relative to external events.

Each of the two capture/compare register arrays contain 16 dual purpose capture/compare registers, each of which may be individually allocated to either CAPCOM timer T0 or T1 (T7 or T8, respectively), and programmed for capture or compare functions. Each of the 32 registers has one associated port pin which serves as an input pin for triggering the capture function, or as an output pin to indicate the occurrence of a compare event.

When a capture/compare register has been selected for capture mode, the current contents of the allocated timer will be latched (captured) into the capture/compare register in response to an external event at the port pin which is associated with this register. In addition, a specific interrupt request for this capture/compare register is generated.

Either a positive, a negative, or both a positive and a negative external signal transition at the pin can be selected as the triggering event. The contents of all registers which have been selected for one of the five compare modes are continuously compared with the contents of the allocated timers.

When a match occurs between the timer value and the value in a capture / compare register, specific actions will be taken based on the selected compare mode.

The input frequencies f_{Tx} , for the timer input selector Tx, are determined as a function of the CPU clocks. The timer input frequencies, resolution and periods which result from the selected pre-scaler option in TxI when using a 40 MHz and 64 MHz CPU clock are listed in the [Table 44](#) and [Table 45](#) respectively.

The numbers for the timer periods are based on a reload value of 0000h. Note that some numbers may be rounded to 3 significant figures.

Table 43. Compare modes

| Compare Modes | Function |
|----------------------|--|
| Mode 0 | Interrupt-only compare mode; several compare interrupts per timer period are possible |
| Mode 1 | Pin toggles on each compare match; several compare events per timer period are possible |
| Mode 2 | Interrupt-only compare mode; only one compare interrupt per timer period is generated |
| Mode 3 | Pin set '1' on match; pin reset '0' on compare time overflow; only one compare event per timer period is generated |
| Double Register Mode | Two registers operate on one pin; pin toggles on each compare match; several compare events per timer period are possible. |

Table 44. CAPCOM timer input frequencies, resolutions and periods at 40 MHz

| $f_{CPU} = 40 \text{ MHz}$ | Timer Input Selection TxI | | | | | | | |
|----------------------------|---------------------------|--------|---------|----------|-----------|------------|------------|----------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler for f_{CPU} | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| Input Frequency | 5MHz | 2.5MHz | 1.25MHz | 625 kHz | 312.5 kHz | 156.25 kHz | 78.125 kHz | 39.1 kHz |
| Resolution | 200ns | 400ns | 0.8µs | 1.6µs | 3.2µs | 6.4µs | 12.8µs | 25.6µs |
| Period | 13.1ms | 26.2ms | 52.4ms | 104.8 ms | 209.7ms | 419.4ms | 838.9ms | 1.678s |

Table 45. CAPCOM timer input frequencies, resolutions and periods at 64 MHz

| $f_{CPU} = 64 \text{ MHz}$ | Timer Input Selection TxI | | | | | | | |
|----------------------------|---------------------------|--------|--------|--------|---------|---------|---------|--------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler for f_{CPU} | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| Input Frequency | 8MHz | 4MHz | 2MHz | 1 kHz | 500 kHz | 250 kHz | 128 kHz | 64 kHz |
| Resolution | 125ns | 250ns | 0.5µs | 1.0µs | 2.0µs | 4.0µs | 8.0µs | 16.0µs |
| Period | 8.2ms | 16.4ms | 32.8ms | 65.5ms | 131.1ms | 262.1ms | 524.3ms | 1.049s |

10 General purpose timer unit

The GPT unit is a flexible multifunctional timer/counter structure which is used for time related tasks such as event timing and counting, pulse width and duty cycle measurements, pulse generation, or pulse multiplication. The GPT unit contains five 16-bit timers organized into two separate modules GPT1 and GPT2. Each timer in each module may operate independently in several different modes, or may be concatenated with another timer of the same module.

10.1 GPT1

Each of the three timers T2, T3, T4 of the GPT1 module can be configured individually for one of four basic modes of operation: **timer, gated timer, counter mode and incremental interface mode**.

In timer mode, the input clock for a timer is derived from the CPU clock, divided by a programmable prescaler.

In counter mode, the timer is clocked in reference to external events.

Pulse width or duty cycle measurement is supported in gated timer mode where the operation of a timer is controlled by the 'gate' level on an external input pin. For these purposes, each timer has one associated port pin (TxIN) which serves as gate or clock input.

Table 46 and Table 47 list the timer input frequencies, resolution and periods for each pre-scaler option at 40MHz and 64MHz CPU clock respectively.

In Incremental Interface Mode, the GPT1 timers (T2, T3, T4) can be directly connected to the incremental position sensor signals A and B by their respective inputs TxIN and TxEUD.

Direction and count signals are internally derived from these two input signals so that the contents of the respective timer Tx corresponds to the sensor position. The third position sensor signal TOP0 can be connected to an interrupt input.

Timer T3 has output toggle latches (TxOTL) which changes state on each timer over flow / underflow. The state of this latch may be output on port pins (TxOUT) for time out monitoring of external hardware components, or may be used internally to clock timers T2 and T4 for high resolution of long duration measurements.

In addition to their basic operating modes, timers T2 and T4 may be configured as reload or capture registers for timer T3.

Table 46. GPT1 timer input frequencies, resolutions and periods at 40 MHz

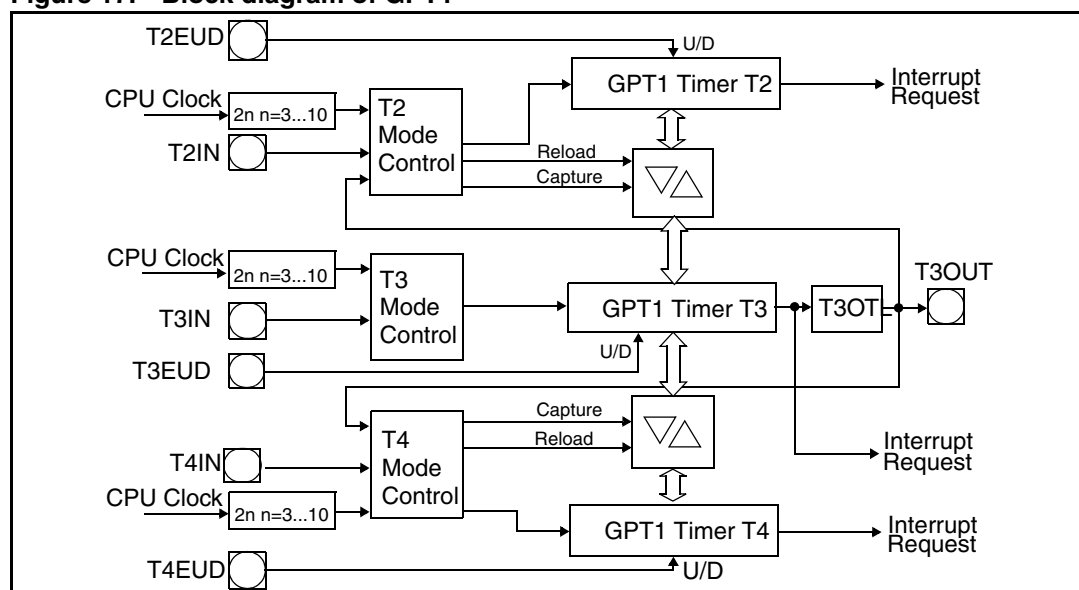
| $f_{CPU} = 40 \text{ MHz}$ | Timer Input Selection T2I / T3I / T4I | | | | | | | |
|----------------------------|---------------------------------------|--------|----------|---------|-----------|------------|------------|----------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler factor | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| Input frequency | 5MHz | 2.5MHz | 1.25 MHz | 625 kHz | 312.5 kHz | 156.25 kHz | 78.125 kHz | 39.1 kHz |

Table 46. GPT1 timer input frequencies, resolutions and periods at 40 MHz

| $f_{\text{CPU}} = 40 \text{ MHz}$ | Timer Input Selection T2I / T3I / T4I | | | | | | | |
|-----------------------------------|---------------------------------------|--------|--------|----------|---------|---------|---------|--------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Resolution | 200ns | 400ns | 0.8µs | 1.6µs | 3.2µs | 6.4µs | 12.8µs | 25.6µs |
| Period maximum | 13.1ms | 26.2ms | 52.4ms | 104.8 ms | 209.7ms | 419.4ms | 838.9ms | 1.678s |

Table 47. GPT1 timer input frequencies, resolutions and periods at 64 MHz

| $f_{\text{CPU}} = 64 \text{ MHz}$ | Timer Input Selection T2I / T3I / T4I | | | | | | | |
|-----------------------------------|---------------------------------------|--------|--------|--------|---------|---------|---------|--------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler factor | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 |
| Input Freq | 8MHz | 4MHz | 2MHz | 1 kHz | 500 kHz | 250 kHz | 128 kHz | 64 kHz |
| Resolution | 125ns | 250ns | 0.5µs | 1.0µs | 2.0µs | 4.0µs | 8.0µs | 16.0µs |
| Period maximum | 8.2ms | 16.4ms | 32.8ms | 65.5ms | 131.1ms | 262.1ms | 524.3ms | 1.049s |

Figure 17. Block diagram of GPT1

10.2 GPT2

The GPT2 module provides precise event control and time measurement. It includes two timers (T5, T6) and a capture/reload register (CAPREL). Both timers can be clocked with an input clock which is derived from the CPU clock via a programmable prescaler or with external signals. The count direction (up/down) for each timer is programmable by software or may additionally be altered dynamically by an external signal on a port pin (TxEUD). Concatenation of the timers is supported via the output toggle latch (T6OTL) of timer T6 which changes its state on each timer overflow/underflow.

The state of this latch may be used to clock timer T5, or it may be output on a port pin (T6OUT). The overflow / underflow of timer T6 can additionally be used to clock the CAPCOM timers T0 or T1, and to cause a reload from the CAPREL register. The CAPREL register may capture the contents of timer T5 based on an external signal transition on the corresponding port pin (CAPIN), and timer T5 may optionally be cleared after the capture procedure. This allows absolute time differences to be measured or pulse multiplication to be performed without software overhead.

The capture trigger (timer T5 to CAPREL) may also be generated upon transitions of GPT1 timer T3 inputs T3IN and/or T3EUD. This is advantageous when T3 operates in Incremental Interface Mode.

[Table 48](#) and [Table 49](#) list the timer input frequencies, resolution and periods for each pre-scaler option at 40MHz and 64MHz CPU clock respectively.

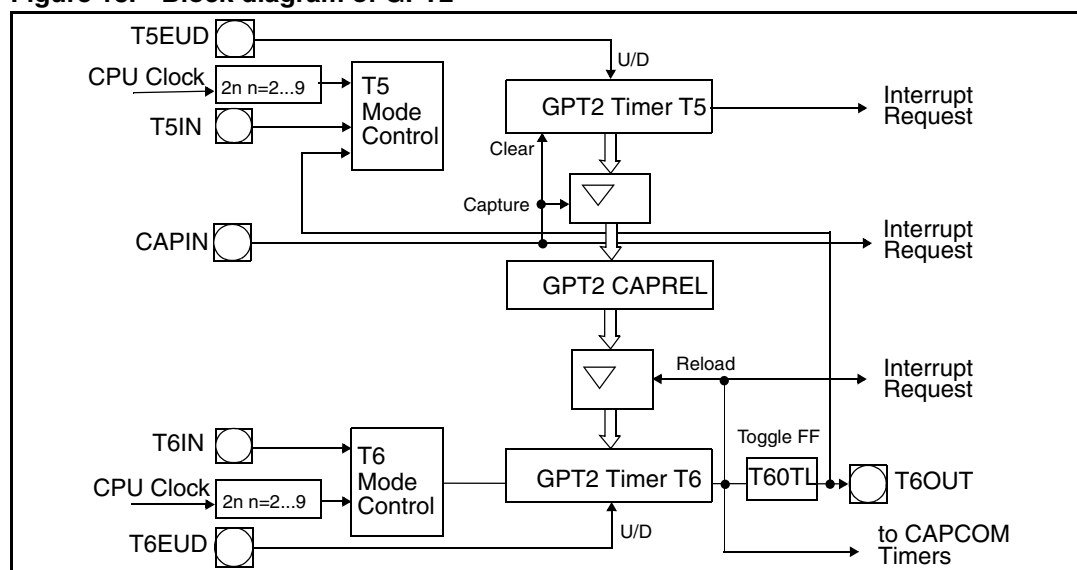
Table 48. GPT2 timer input frequencies, resolutions and periods at 40 MHz

| $f_{CPU} = 40\text{MHz}$ | Timer Input Selection T5I / T6I | | | | | | | |
|--------------------------|---------------------------------|--------|--------|-------------|-------------|-------------|-------------|--------------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler factor | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
| Input Freq | 10MHz | 5MHz | 2.5MHz | 1.25 MHz | 625 kHz | 312.5 kHz | 156.25 kHz | 78.125 kHz |
| Resolution | 100ns | 200ns | 400ns | 0.8 μ s | 1.6 μ s | 3.2 μ s | 6.4 μ s | 12.8 μ s |
| Period maximum | 6.55ms | 13.1ms | 26.2ms | 52.4ms | 104.8ms | 209.7ms | 419.4ms | 838.9ms |

Table 49. GPT2 timer input frequencies, resolutions and periods at 64 MHz

| $f_{CPU} = 64\text{MHz}$ | Timer Input Selection T5I / T6I | | | | | | | |
|--------------------------|---------------------------------|-------|--------|-------------|-------------|-------------|-------------|-------------|
| | 000b | 001b | 010b | 011b | 100b | 101b | 110b | 111b |
| Pre-scaler factor | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 |
| Input Freq | 16MHz | 8MHz | 4MHz | 2MHz | 1 kHz | 500 kHz | 250 kHz | 128 kHz |
| Resolution | 62.5ns | 125ns | 250ns | 0.5 μ s | 1.0 μ s | 2.0 μ s | 4.0 μ s | 8.0 μ s |
| Period maximum | 4.1ms | 8.2ms | 16.4ms | 32.8ms | 65.5ms | 131.1ms | 262.1ms | 524.3ms |

Figure 18. Block diagram of GPT2



11 PWM modules

Two pulse width modulation modules are available on ST10F276: standard PWM0 and XBUS PWM1. They can generate up to four PWM output signals each, using edge-aligned or centre-aligned PWM. In addition, the PWM modules can generate PWM burst signals and single shot outputs. The [Table 50](#) and [Table 51](#) show the PWM frequencies for different resolutions. The level of the output signals is selectable and the PWM modules can generate interrupt requests.

Figure 19. Block diagram of PWM module

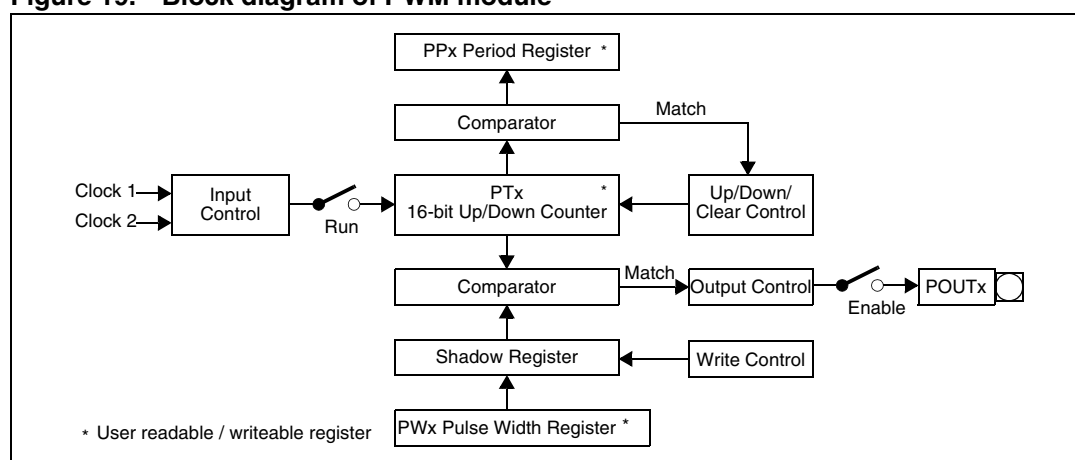


Table 50. PWM unit frequencies and resolutions at 40 MHz CPU clock

| Mode 0 | Resolution | 8-bit | 10-bit | 12-bit | 14-bit | 16-bit |
|--------------|------------|------------|-----------|----------|----------|---------|
| CPU Clock/1 | 25ns | 156.25 kHz | 39.1 kHz | 9.77 kHz | 2.44Hz | 610Hz |
| CPU Clock/64 | 1.6µs | 2.44 kHz | 610Hz | 152.6Hz | 38.15Hz | 9.54Hz |
| Mode 1 | Resolution | 8-bit | 10-bit | 12-bit | 14-bit | 16-bit |
| CPU Clock/1 | 25ns | 78.12 kHz | 19.53 kHz | 4.88 kHz | 1.22 kHz | 305.2Hz |
| CPU Clock/64 | 1.6µs | 1.22 kHz | 305.17Hz | 76.29Hz | 19.07Hz | 4.77Hz |

Table 51. PWM unit frequencies and resolutions at 64 MHz CPU clock

| Mode 0 | Resolution | 8-bit | 10-bit | 12-bit | 14-bit | 16-bit |
|--------------|------------|----------|-----------|-----------|----------|---------|
| CPU Clock/1 | 15.6ns | 250 kHz | 62.5 kHz | 15.63 kHz | 3.91Hz | 977Hz |
| CPU Clock/64 | 1.0µs | 3.91 kHz | 976.6Hz | 244.1Hz | 61.01Hz | 15.26Hz |
| Mode 1 | Resolution | 8-bit | 10-bit | 12-bit | 14-bit | 16-bit |
| CPU Clock/1 | 15.6ns | 125 kHz | 31.25 kHz | 7.81 kHz | 1.95 kHz | 488.3Hz |
| CPU Clock/64 | 1.0µs | 1.95 kHz | 488.28Hz | 122.07Hz | 30.52Hz | 7.63Hz |

12 Parallel ports

12.1 Introduction

The ST10F276 MCU provides up to 111 I/O lines with programmable features. These capabilities bring very flexible adaptation of this MCU to wide range of applications.

ST10F276 has nine groups of I/O lines gathered as follows:

- Port 0 is a two time 8-bit port named P0L (Low as less significant byte) and P0H (high as most significant byte)
- Port 1 is a two time 8-bit port named P1L and P1H
- Port 2 is a 16-bit port
- Port 3 is a 15-bit port (P3.14 line is not implemented)
- Port 4 is a 8-bit port
- Port 5 is a 16-bit port input only
- Port 6, Port 7 and Port 8 are 8-bit ports

These ports may be used as general purpose bidirectional input or output, software controlled with dedicated registers.

For example, the output drivers of six of the ports (2, 3, 4, 6, 7, 8) can be configured (bit-wise) for push-pull or open drain operation using ODPx registers.

The input threshold levels are programmable (TTL/CMOS) for all the ports. The logic level of a pin is clocked into the input latch once per state time, regardless whether the port is configured for input or output. The threshold is selected with PICON and XPICON registers control bits.

A write operation to a port pin configured as an input causes the value to be written into the port output latch, while a read operation returns the latched state of the pin itself. A read-modify-write operation reads the value of the pin, modifies it, and writes it back to the output latch.

Writing to a pin configured as an output (DPx.y='1') causes the output latch and the pin to have the written value, since the output buffer is enabled. Reading this pin returns the value of the output latch. A read-modify-write operation reads the value of the output latch, modifies it, and writes it back to the output latch, thus also modifying the level at the pin.

I/O lines support an alternate function which is detailed in the following description of each port.

12.2 I/O's special features

12.2.1 Open drain mode

Some of the I/O ports of ST10F276 support the open drain capability. This programmable feature may be used with an external pull-up resistor, in order to get an AND wired logical function.

This feature is implemented for ports P2, P3, P4, P6, P7 and P8 (see respective sections), and is controlled through the respective Open Drain Control Registers ODPx.

12.2.2 Input threshold control

The standard inputs of the ST10F276 determine the status of input signals according to TTL levels. In order to accept and recognize noisy signals, CMOS input thresholds can be selected instead of the standard TTL thresholds for all the pins. These CMOS thresholds are defined above the TTL thresholds and feature a higher hysteresis to prevent the inputs from toggling while the respective input signal level is near the thresholds.

The Port Input Control registers PICON and XPICON are used to select these thresholds for each Byte of the indicated ports, this means the 8-bit ports P0L, P0H, P1L, P1H, P4, P7 and P8 are controlled by one bit each while ports P2, P3 and P5 are controlled by two bits each.

All options for individual direction and output mode control are available for each pin, independent of the selected input threshold.

12.3 Alternate port functions

Each port line has one associated programmable alternate input or output function.

- PORT0 and PORT1 may be used as address and data lines when accessing external memory. Besides, PORT1 provides also:
 - Input capture lines
 - 8 additional analog input channels to the A/D converter
- Port 2, Port 7 and Port 8 are associated with the capture inputs or compare outputs of the CAPCOM units and/or with the outputs of the PWM0 module, of the PWM1 module and of the ASC1.
Port 2 is also used for fast external interrupt inputs and for timer 7 input.
- Port 3 includes the alternate functions of timers, serial interfaces, the optional bus control signal $\overline{\text{BHE}}$ and the system clock output (CLKOUT).
- Port 4 outputs the additional segment address bit A23...A16 in systems where more than 64 Kbytes of memory are to be access directly. In addition, CAN1, CAN2 and I²C lines are provided.
- Port 5 is used as analog input channels of the A/D converter or as timer control signals.
- Port 6 provides optional bus arbitration signals ($\overline{\text{BREQ}}$, $\overline{\text{HLDA}}$, $\overline{\text{HOLD}}$) and chip select signals and the SSC1 lines.

If the alternate output function of a pin is to be used, the direction of this pin must be programmed for output (DPx.y='1'), except for some signals that are used directly after reset and are configured automatically. Otherwise the pin remains in the high-impedance state and is not effected by the alternate output function. The respective port latch should hold a

'1', because its output is ANDed with the alternate output data (except for PWM output signals).

If the alternate input function of a pin is used, the direction of the pin must be programmed for input (DPx.y='0') if an external device is driving the pin. The input direction is the default after reset. If no external device is connected to the pin, however, one can also set the direction for this pin to output. In this case, the pin reflects the state of the port output latch. Thus, the alternate input function reads the value stored in the port output latch. This can be used for testing purposes to allow a software trigger of an alternate input function by writing to the port output latch.

On most of the port lines, the user software is responsible for setting the proper direction when using an alternate input or output function of a pin.

This is done by setting or clearing the direction control bit DPx.y of the pin before enabling the alternate function.

There are port lines, however, where the direction of the port line is switched automatically.

For instance, in the multiplexed external bus modes of PORT0, the direction must be switched several times for an instruction fetch in order to output the addresses and to input the data.

Obviously, this cannot be done through instructions. In these cases, the direction of the port line is switched automatically by hardware if the alternate function of such a pin is enabled.

To determine the appropriate level of the port output latches check how the alternate data output is combined with the respective port latch output.

There is one basic structure for all port lines with only an alternate input function. Port lines with only an alternate output function, however, have different structures due to the way the direction of the pin is switched and depending on whether the pin is accessible by the user software or not in the alternate function mode.

All port lines that are not used for these alternate functions may be used as general purpose I/O lines.

13 A/D converter

A 10-bit A/D converter with 16+8 multiplexed input channels and a sample and hold circuit is integrated on-chip. An automatic self-calibration adjusts the A/D converter module to process parameter variations at each reset event. The sample time (for loading the capacitors) and the conversion time is programmable and can be adjusted to the external circuitry.

The ST10F273E has 16+8 multiplexed input channels on Port 5 and Port 1. The selection between Port 5 and Port 1 is made via a bit in a XBus register. Refer to the User Manual for a detailed description.

A different accuracy is guaranteed (Total Unadjusted Error) on Port 5 and Port 1 analog channels (with higher restrictions when overload conditions occur); in particular, Port 5 channels are more accurate than the Port 1 ones. Refer to Electrical Characteristic section for details.

The A/D converter input bandwidth is limited by the achievable accuracy: supposing a maximum error of 0.5LSB (2mV) impacting the global TUE (TUE depends also on other causes), in worst case of temperature and process, the maximum frequency for a sine wave analog signal is around 7.5 kHz. Of course, to reduce the effect of the input signal variation on the accuracy down to 0.05LSB, the maximum input frequency of the sine wave shall be reduced to 800 Hz.

If static signal is applied during sampling phase, series resistance shall not be greater than 20k Ω (this taking into account eventual input leakage). It is suggested to not connect any capacitance on analog input pins, in order to reduce the effect of charge partitioning (and consequent voltage drop error) between the external and the internal capacitance: in case an RC filter is necessary the external capacitance must be greater than 10nF to minimize the accuracy impact.

Overrun error detection / protection is controlled by the ADDAT register. Either an interrupt request is generated when the result of a previous conversion has not been read from the result register at the time the next conversion is complete, or the next conversion is suspended until the previous result has been read. For applications which require less than 16+8 analog input channels, the remaining channel inputs can be used as digital input port pins.

The A/D converter of the ST10F276 supports different conversion modes:

- **Single channel single conversion:** The analog level of the selected channel is sampled once and converted. The result of the conversion is stored in the ADDAT register.
- **Single channel continuous conversion:** The analog level of the selected channel is repeatedly sampled and converted. The result of the conversion is stored in the ADDAT register.
- **Auto scan single conversion:** The analog level of the selected channels are sampled once and converted. After each conversion the result is stored in the ADDAT register. The data can be transferred to the RAM by interrupt software management or using the powerful Peripheral Event Controller (PEC) data transfer.
- **Auto scan continuous conversion:** The analog level of the selected channels are repeatedly sampled and converted. The result of the conversion is stored in the ADDAT

register. The data can be transferred to the RAM by interrupt software management or using the PEC data transfer.

- **Wait for ADDAT read mode:** When using continuous modes, in order to avoid to overwrite the result of the current conversion by the next one, the ADWR bit of ADCON control register must be activated. Then, until the ADDAT register is read, the new result is stored in a temporary buffer and the conversion is on hold.
- **Channel injection mode:** When using continuous modes, a selected channel can be converted in between without changing the current operating mode. The 10-bit data of the conversion are stored in ADRES field of ADDAT2. The current continuous mode remains active after the single conversion is completed.

A full calibration sequence is performed after a reset. This full calibration lasts up to 40.630 CPU clock cycles. During this time, the busy flag ADBSY is set to indicate the operation. It compensates the capacitance mismatch, so the calibration procedure does not need any update during normal operation.

No conversion can be performed during this time: the bit ADBSY shall be polled to verify when the calibration is over, and the module is able to start a conversion.

14 Serial channels

Serial communication with other microcontrollers, microprocessors, terminals or external peripheral components is provided by up to four serial interfaces: two asynchronous / synchronous serial channels (ASC0 and ASC1) and two high-speed synchronous serial channel (SSC0 and SSC1). Dedicated Baud rate generators set up all standard Baud rates without the requirement of oscillator tuning. For transmission, reception and erroneous reception, separate interrupt vectors are provided for ASC0 and SSC0 serial channel. A more complex mechanism of interrupt sources multiplexing is implemented for ASC1 and SSC1 (XBUS mapped).

14.1 Asynchronous / synchronous serial interfaces

The asynchronous / synchronous serial interfaces (ASC0 and ASC1) provides serial communication between the ST10F276 and other microcontrollers, microprocessors or external peripherals.

14.2 ASCx in asynchronous mode

In asynchronous mode, 8- or 9-bit data transfer, parity generation and the number of stop bits can be selected. Parity framing and overrun error detection is provided to increase the reliability of data transfers. Transmission and reception of data is double-buffered. Full-duplex communication up to 2M Bauds (at 64 MHz of f_{CPU}) is supported in this mode.

Table 52. ASC asynchronous baud rates by reload value and deviation errors ($f_{CPU} = 40$ MHz)

| S0BRS = '0', $f_{CPU} = 40$ MHz | | | S0BRS = '1', $f_{CPU} = 40$ MHz | | |
|---------------------------------|-----------------|--------------------|---------------------------------|-----------------|--------------------|
| Baud Rate (Baud) | Deviation Error | Reload Value (hex) | Baud Rate (Baud) | Deviation Error | Reload Value (hex) |
| 1 250 000 | 0.0% / 0.0% | 0000 / 0000 | 833 333 | 0.0% / 0.0% | 0000 / 0000 |
| 112 000 | +1.5% / -7.0% | 000A / 000B | 112 000 | +6.3% / -7.0% | 0006 / 0007 |
| 56 000 | +1.5% / -3.0% | 0015 / 0016 | 56 000 | +6.3% / -0.8% | 000D / 000E |
| 38 400 | +1.7% / -1.4% | 001F / 0020 | 38 400 | +3.3% / -1.4% | 0014 / 0015 |
| 19 200 | +0.2% / -1.4% | 0040 / 0041 | 19 200 | +0.9% / -1.4% | 002A / 002B |
| 9 600 | +0.2% / -0.6% | 0081 / 0082 | 9 600 | +0.9% / -0.2% | 0055 / 0056 |
| 4 800 | +0.2% / -0.2% | 0103 / 0104 | 4 800 | +0.4% / -0.2% | 00AC / 00AD |
| 2 400 | +0.2% / 0.0% | 0207 / 0208 | 2 400 | +0.1% / -0.2% | 015A / 015B |
| 1 200 | 0.1% / 0.0% | 0410 / 0411 | 1 200 | +0.1% / -0.1% | 02B5 / 02B6 |
| 600 | 0.0% / 0.0% | 0822 / 0823 | 600 | +0.1% / 0.0% | 056B / 056C |
| 300 | 0.0% / 0.0% | 1045 / 1046 | 300 | 0.0% / 0.0% | 0AD8 / 0AD9 |
| 153 | 0.0% / 0.0% | 1FE8 / 1FE9 | 102 | 0.0% / 0.0% | 1FE8 / 1FE9 |

Table 53. ASC asynchronous baud rates by reload value and deviation errors ($f_{CPU} = 64$ MHz)

| S0BRS = '0', $f_{CPU} = 64$ MHz | | | S0BRS = '1', $f_{CPU} = 64$ MHz | | |
|---------------------------------|-----------------|--------------------|---------------------------------|-----------------|--------------------|
| Baud Rate (Baud) | Deviation Error | Reload Value (hex) | Baud Rate (Baud) | Deviation Error | Reload Value (hex) |
| 2 000 000 | 0.0% / 0.0% | 0000 / 0000 | 1 333 333 | 0.0% / 0.0% | 0000 / 0000 |
| 112 000 | +1.5% / -7.0% | 0010 / 0011 | 112 000 | +6.3% / -7.0% | 000A / 000B |
| 56 000 | +1.5% / -3.0% | 0022 / 0023 | 56 000 | +6.3% / -0.8% | 0016 / 0017 |
| 38 400 | +1.7% / -1.4% | 0033 / 0034 | 38 400 | +3.3% / -1.4% | 0021 / 0022 |
| 19 200 | +0.2% / -1.4% | 0067 / 0068 | 19 200 | +0.9% / -1.4% | 0044 / 0045 |
| 9 600 | +0.2% / -0.6% | 00CF / 00D0 | 9 600 | +0.9% / -0.2% | 0089 / 008A |
| 4 800 | +0.2% / -0.2% | 019F / 01A0 | 4 800 | +0.4% / -0.2% | 0114 / 0115 |
| 2 400 | +0.2% / 0.0% | 0340 / 0341 | 2 400 | +0.1% / -0.2% | 022A / 015B |
| 1 200 | 0.1% / 0.0% | 0681 / 0682 | 1 200 | +0.1% / -0.1% | 0456 / 0457 |
| 600 | 0.0% / 0.0% | 0D04 / 0D05 | 600 | +0.1% / 0.0% | 08AD / 08AE |
| 300 | 0.0% / 0.0% | 1A09 / 1A0A | 300 | 0.0% / 0.0% | 115B / 115C |
| 245 | 0.0% / 0.0% | 1FE2 / 1FE3 | 163 | 0.0% / 0.0% | 1FF2 / 1FF3 |

Note: The deviation errors given in the [Table 52](#) and [Table 53](#) are rounded. To avoid deviation errors use a Baud rate crystal (providing a multiple of the ASC0 sampling frequency).

14.3 ASCx in synchronous mode

In synchronous mode, data is transmitted or received synchronously to a shift clock which is generated by the ST10F276. Half-duplex communication up to 8M Baud (at 40 MHz of f_{CPU}) is possible in this mode.

Table 54. ASC synchronous baud rates by reload value and deviation errors ($f_{CPU} = 40$ MHz)

| S0BRS = '0', $f_{CPU} = 40$ MHz | | | S0BRS = '1', $f_{CPU} = 40$ MHz | | |
|---------------------------------|-----------------|--------------------|---------------------------------|-----------------|--------------------|
| Baud Rate (Baud) | Deviation Error | Reload Value (hex) | Baud Rate (Baud) | Deviation Error | Reload Value (hex) |
| 5 000 000 | 0.0% / 0.0% | 0000 / 0000 | 3 333 333 | 0.0% / 0.0% | 0000 / 0000 |
| 112 000 | +1.5% / -0.8% | 002B / 002C | 112 000 | +2.6% / -0.8% | 001C / 001D |
| 56 000 | +0.3% / -0.8% | 0058 / 0059 | 56 000 | +0.9% / -0.8% | 003A / 003B |
| 38 400 | +0.2% / -0.6% | 0081 / 0082 | 38 400 | +0.9% / -0.2% | 0055 / 0056 |
| 19 200 | +0.2% / -0.2% | 0103 / 0104 | 19 200 | +0.4% / -0.2% | 00AC / 00AD |
| 9 600 | +0.2% / 0.0% | 0207 / 0208 | 9 600 | +0.1% / -0.2% | 015A / 015B |
| 4 800 | +0.1% / 0.0% | 0410 / 0411 | 4 800 | +0.1% / -0.1% | 02B5 / 02B6 |
| 2 400 | 0.0% / 0.0% | 0822 / 0823 | 2 400 | +0.1% / 0.0% | 056B / 056C |
| 1 200 | 0.0% / 0.0% | 1045 / 1046 | 1 200 | 0.0% / 0.0% | 0AD8 / 0AD9 |

Table 54. ASC synchronous baud rates by reload value and deviation errors ($f_{CPU} = 40$ MHz)

| S0BRS = '0', $f_{CPU} = 40$ MHz | | | S0BRS = '1', $f_{CPU} = 40$ MHz | | |
|---------------------------------|-----------------|--------------------|---------------------------------|-----------------|--------------------|
| Baud Rate (Baud) | Deviation Error | Reload Value (hex) | Baud Rate (Baud) | Deviation Error | Reload Value (hex) |
| 900 | 0.0% / 0.0% | 15B2 / 15B3 | 600 | 0.0% / 0.0% | 15B2 / 15B3 |
| 612 | 0.0% / 0.0% | 1FE8 / 1FE9 | 407 | 0.0% / 0.0% | 1FFD / 1FFE |

Table 55. ASC synchronous baud rates by reload value and deviation errors ($f_{CPU} = 64$ MHz)

| S0BRS = '0', $f_{CPU} = 64$ MHz | | | S0BRS = '1', $f_{CPU} = 64$ MHz | | |
|---------------------------------|-----------------|--------------------|---------------------------------|-----------------|--------------------|
| Baud Rate (Baud) | Deviation Error | Reload Value (hex) | Baud Rate (Baud) | Deviation Error | Reload Value (hex) |
| 8 000 000 | 0.0% / 0.0% | 0000 / 0000 | 5 333 333 | 0.0% / 0.0% | 0000 / 0000 |
| 112 000 | +0.6% / -0.8% | 0046 / 0047 | 112 000 | +1.3% / -0.8% | 002E / 002F |
| 56 000 | +0.6% / -0.1% | 008D / 008E | 56 000 | +0.3% / -0.8% | 005E / 005F |
| 38 400 | +0.2% / -0.3% | 00CF / 00D0 | 38 400 | +0.6% / -0.1% | 0089 / 008A |
| 19 200 | +0.2% / -0.1% | 019F / 01A0 | 19 200 | +0.3% / -0.1% | 0114 / 0115 |
| 9 600 | +0.0% / -0.1% | 0340 / 0341 | 9 600 | +0.1% / -0.1% | 022A / 022B |
| 4 800 | 0.0% / 0.0% | 0681 / 0682 | 4 800 | 0.0% / -0.1% | 0456 / 0457 |
| 2 400 | 0.0% / 0.0% | 0D04 / 0D05 | 2 400 | 0.0% / 0.0% | 08AD / 08AE |
| 1 200 | 0.0% / 0.0% | 1A09 / 1A0A | 1 200 | 0.0% / 0.0% | 115B / 115C |
| 977 | 0.0% / 0.0% | 1FFB / 1FFC | 900 | 0.0% / 0.0% | 1724 / 1725 |
| | | | 652 | 0.0% / 0.0% | 1FF2 / 1FF3 |

Note: The deviation errors given in the [Table 54](#) and [Table 55](#) are rounded. To avoid deviation errors use a Baud rate crystal (providing a multiple of the ASC0 sampling frequency)

14.4 High speed synchronous serial interfaces

The High-Speed Synchronous Serial Interfaces (SSC0 and SSC1) provides flexible high-speed serial communication between the ST10F276 and other microcontrollers, microprocessors or external peripherals.

The SSCx supports full-duplex and half-duplex synchronous communication. The serial clock signal can be generated by the SSCx itself (master mode) or be received from an external master (slave mode). Data width, shift direction, clock polarity and phase are programmable.

This allows communication with SPI-compatible devices. Transmission and reception of data is double-buffered. A 16-bit Baud rate generator provides the SSCx with a separate serial clock signal. The serial channel SSCx has its own dedicated 16-bit Baud rate generator with 16-bit reload capability, allowing Baud rate generation independent from the timers.

[Table 56](#) and [Table 57](#) list some possible Baud rates against the required reload values and the resulting bit times for 40 MHz and 64 MHz CPU clock respectively. The maximum is anyway limited to 8Mbaud.

Table 56. Synchronous baud rate and reload values ($f_{\text{CPU}} = 40 \text{ MHz}$)

| Baud Rate | Bit Time | Reload Value |
|--|-------------|--------------|
| Reserved | --- | 0000h |
| Can be used only with $f_{\text{CPU}} = 32 \text{ MHz}$ (or lower) | --- | 0001h |
| 6.6M Baud | 150ns | 0002h |
| 5M Baud | 200ns | 0003h |
| 2.5M Baud | 400ns | 0007h |
| 1M Baud | 1 μ s | 0013h |
| 100K Baud | 10 μ s | 00C7h |
| 10K Baud | 100 μ s | 07CFh |
| 1K Baud | 1ms | 4E1Fh |
| 306 Baud | 3.26ms | FF4Eh |

Table 57. Synchronous baud rate and reload values ($f_{\text{CPU}} = 64 \text{ MHz}$)

| Baud Rate | Bit Time | Reload Value |
|--|-------------|--------------|
| Reserved | --- | 0000h |
| Can be used only with $f_{\text{CPU}} = 32 \text{ MHz}$ (or lower) | --- | 0001h |
| Can be used only with $f_{\text{CPU}} = 48 \text{ MHz}$ (or lower) | --- | 0002h |
| 8M Baud | 125ns | 0003h |
| 4M Baud | 250ns | 0007h |
| 1M Baud | 1 μ s | 001Fh |
| 100K Baud | 10 μ s | 013Fh |
| 10K Baud | 100 μ s | 0C7Fh |
| 1K Baud | 1ms | 7CFFh |
| 489 Baud | 2.04ms | FF9Eh |

15 I2C interface

The integrated I²C Bus Module handles the transmission and reception of frames over the two-line SDA/SCL in accordance with the I²C Bus specification. The I²C Module can operate in slave mode, in master mode or in multi-master mode. It can receive and transmit data using 7-bit or 10-bit addressing. Data can be transferred at speeds up to 400 Kbit/s (both Standard and Fast I²C bus modes are supported).

The module can generate three different types of interrupt:

- Requests related to bus events, like start or stop events, arbitration lost, etc.
- Requests related to data transmission
- Requests related to data reception

These requests are issued to the interrupt controller by three different lines, and identified as Error, Transmit, and Receive interrupt lines.

When the I²C module is enabled by setting bit XI2CEN in XPERCON register, pins P4.4 and P4.7 (where SCL and SDA are respectively mapped as alternate functions) are automatically configured as bidirectional open-drain: the value of the external pull-up resistor depends on the application. P4, DP4 and ODP4 cannot influence the pin configuration.

When the I²C cell is disabled (clearing bit XI2CEN), P4.4 and P4.7 pins are standard I/O controlled by P4, DP4 and ODP4.

The speed of the I²C interface may be selected between Standard mode (0 to 100 kHz) and Fast I²C mode (100 to 400 kHz).

16 CAN modules

The two integrated CAN modules (CAN1 and CAN2) are identical and handle the completely autonomous transmission and reception of CAN frames according to the CAN specification V2.0 part B (active). It is based on the C-CAN specification.

Each on-chip CAN module can receive and transmit standard frames with 11-bit identifiers as well as extended frames with 29-bit identifiers.

Because of duplication of the CAN controllers, the following adjustments are to be considered:

- Same internal register addresses of both CAN controllers, but with base addresses differing in address bit A8; separate chip select for each CAN module. Refer to [Chapter 4: Internal Flash memory](#).
- The CAN1 transmit line (CAN1_TxD) is the alternate function of the Port P4.6 pin and the receive line (CAN1_RxD) is the alternate function of the Port P4.5 pin.
- The CAN2 transmit line (CAN2_TxD) is the alternate function of the Port P4.7 pin and the receive line (CAN2_RxD) is the alternate function of the Port P4.4 pin.
- Interrupt request lines of the CAN1 and CAN2 modules are connected to the XBUS interrupt lines together with other X-Peripherals sharing the four vectors.
- The CAN modules must be selected with corresponding CANxEN bit of XPERCON register before the bit XPEN of SYSCON register is set.
- The reset default configuration is: CAN1 enabled, CAN2 disabled.

Note: If one or both CAN modules is used, Port 4 cannot be programmed to output all 8 segment address lines. Thus, only four segment address lines can be used, reducing the external memory space to 5 Mbytes (1 Mbyte per \overline{CS} line).

16.1 Configuration support

It is possible that both CAN controllers are working on the same CAN bus, supporting together up to 64 message objects. In this configuration, both receive signals and both transmit signals are linked together when using the same CAN transceiver. This configuration is especially supported by providing open drain outputs for the CAN1_TxD and CAN2_TxD signals. The open drain function is controlled with the ODP4 register for port P4: in this way it is possible to connect together P4.4 with P4.5 (receive lines) and P4.6 with P4.7 (transmit lines configured to be configured as Open-Drain).

The user is also allowed to map internally both CAN modules on the same pins P4.5 and P4.6. In this way, P4.4 and P4.7 may be used either as general purpose I/O lines, or used for I²C interface. This is possible by setting bit CANPAR of XMISC register. To access this register it is necessary to set bit XMISCEN of XPERCON register and bit XPEN of SYSCON register.

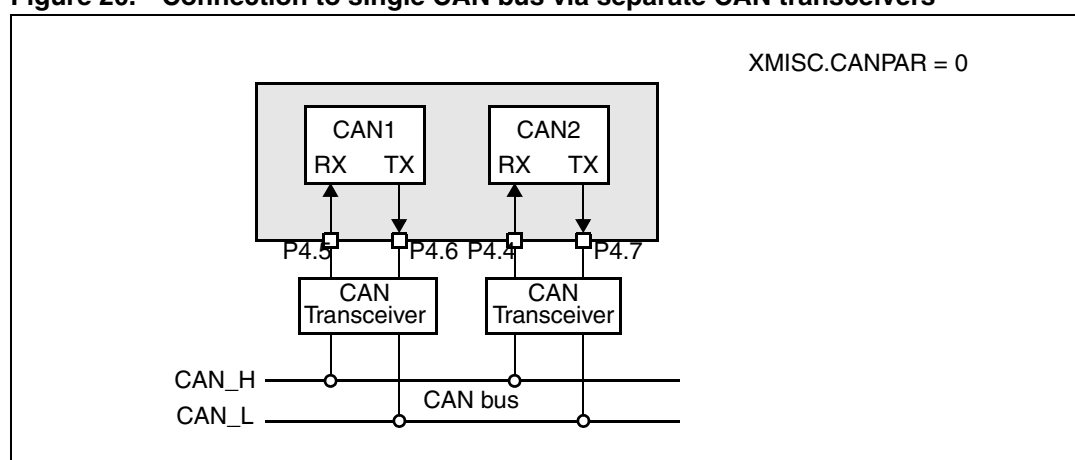
16.2 CAN bus configurations

Depending on application, CAN bus configuration may be one single bus with a single or multiple interfaces or a multiple bus with a single or multiple interfaces. The ST10F276 is able to support these two cases.

Single CAN bus

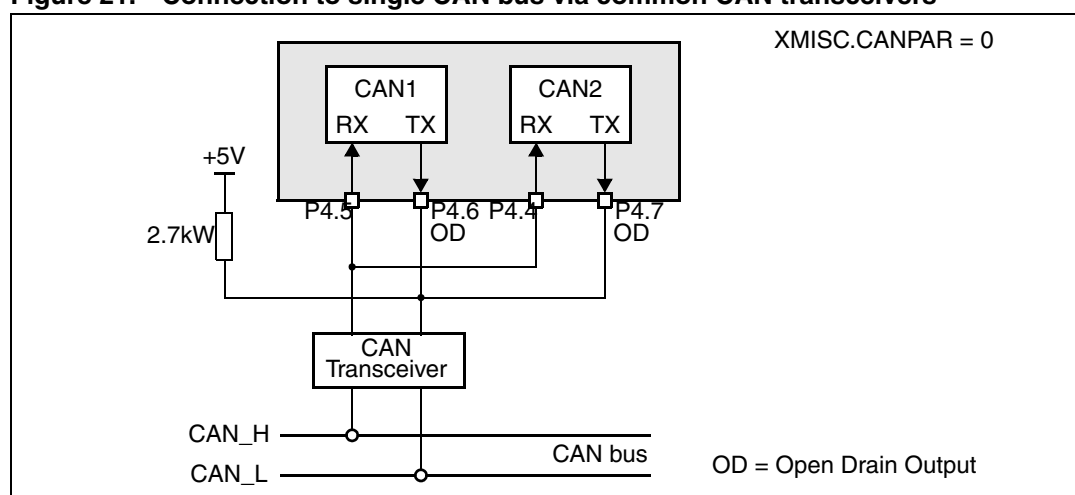
The single CAN Bus multiple interfaces configuration may be implemented using two CAN transceivers as shown in [Figure 20](#).

Figure 20. Connection to single CAN bus via separate CAN transceivers



The ST10F276 also supports single CAN Bus multiple (dual) interfaces using the open drain option of the CANx_TxD output as shown in [Figure 21](#). Thanks to the OR-Wired Connection, only one transceiver is required. In this case the design of the application must take in account the wire length and the noise environment.

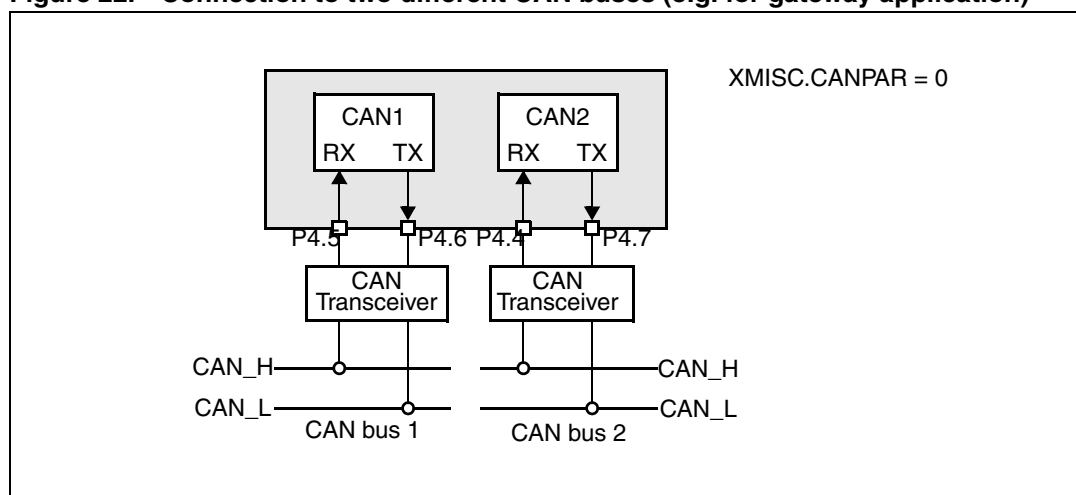
Figure 21. Connection to single CAN bus via common CAN transceivers



Multiple CAN bus

The ST10F276 provides two CAN interfaces to support such kind of bus configuration as shown in [Figure 22](#).

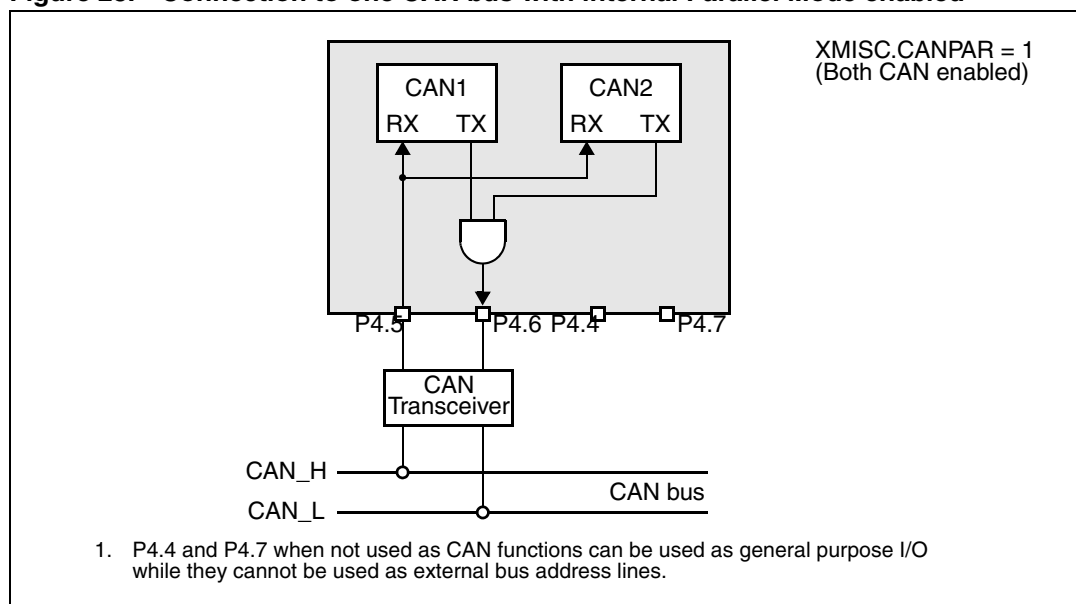
Figure 22. Connection to two different CAN buses (e.g. for gateway application)



Parallel Mode

In addition to previous configurations, a parallel mode is supported. This is shown in [Figure 23](#).

Figure 23. Connection to one CAN bus with internal Parallel Mode enabled



17 Real time clock

The Real Time Clock is an independent timer, in which the clock is derived directly from the clock oscillator on XTAL1 (main oscillator) input or XTAL3 input (32 kHz low-power oscillator) so that it can be kept on running even in Idle or Power down mode (if enabled to). Registers access is implemented onto the XBUS. This module is designed with the following characteristics:

- Generation of the current time and date for the system
- Cyclic time based interrupt, on Port2 external interrupts every 'RTC basic clock tick' and after n 'RTC basic clock ticks' (n is programmable) if enabled
- 58-bit timer for long term measurement
- Capability to exit the ST10 chip from Power down mode (if PWDCFG of SYSCON set) after a programmed delay

The real time clock is based on two main blocks of counters. The first block is a prescaler which generates a basic reference clock (for example a 1 second period). This basic reference clock is coming out of a 20-bit DIVIDER. This 20-bit counter is driven by an input clock derived from the on-chip CPU clock, pre-divided by a 1/64 fixed counter. This 20-bit counter is loaded at each basic reference clock period with the value of the 20-bit PRESCALER register. The value of the 20-bit RTCP register determines the period of the basic reference clock.

A timed interrupt request (RTCSI) may be sent on each basic reference clock period. The second block of the RTC is a 32-bit counter that may be initialized with the current system time. This counter is driven with the basic reference clock signal. In order to provide an alarm function the contents of the counter is compared with a 32-bit alarm register. The alarm register may be loaded with a reference date. An alarm interrupt request (RTCAI), may be generated when the value of the counter matches the alarm register.

The timed RTCSI and the alarm RTCAI interrupt requests can trigger a fast external interrupt via EXISEL register of port 2 and wake-up the ST10 chip when running power down mode. Using the RTCOFF bit of RTCCON register, the user may switch off the clock oscillator when entering the power down mode.

The last function implemented in the RTC is to switch off the main on-chip oscillator and the 32 kHz on chip oscillator if the ST10 enters the Power Down mode, so that the chip can be fully switched off (if RTC is disabled).

At power on, and after Reset phase, if the presence of a 32 kHz oscillation on XTAL3 / XTAL4 pins is detected, then the RTC counter is driven by this low frequency reference clock: when Power Down mode is entered, the RTC can either be stopped or left running, and in both the cases the main oscillator is turned off, reducing the power consumption of the device to the minimum required to keep on running the RTC counter and relative reference oscillator. This is valid also if Stand-by mode is entered (switching off the main supply V_{DD}), since both the RTC and the low power oscillator (32 kHz) are biased by the V_{STBY} . Vice versa, when at power on and after Reset, the 32 kHz is not present, the main oscillator drives the RTC counter, and since it is powered by the main power supply, it cannot be maintained running in Stand-by mode, while in Power Down mode the main oscillator is maintained running to provide the reference to the RTC module (if not disabled).

18 Watchdog timer

The Watchdog Timer is a fail-safe mechanism which prevents the microcontroller from malfunctioning for long periods of time.

The Watchdog Timer is always enabled after a reset of the chip and can only be disabled in the time interval until the EINIT (end of initialization) instruction has been executed.

Therefore, the chip start-up procedure is always monitored. The software must be designed to service the watchdog timer before it overflows. If, due to hardware or software related failures, the software fails to do so, the watchdog timer overflows and generates an internal hardware reset. It pulls the $\overline{\text{RSTOUT}}$ pin low in order to allow external hardware components to be reset.

Each of the different reset sources is indicated in the WDTCON register:

- Watchdog Timer Reset in case of an overflow
- Software Reset in case of execution of the SRST instruction
- Short, Long and Power-On Reset in case of hardware reset (and depending of reset pulse duration and RPD pin configuration)

The indicated bits are cleared with the EINIT instruction. The source of the reset can be identified during the initialization phase.

The Watchdog Timer is 16-bit, clocked with the system clock divided by 2 or 128. The high Byte of the watchdog timer register can be set to a pre-specified reload value (stored in WDTREL).

Each time it is serviced by the application software, the high byte of the watchdog timer is reloaded. For security, rewrite WDTCON each time before the watchdog timer is serviced

The [Table 58](#) and [Table 59](#) show the watchdog time range for 40 MHz and 64 MHz CPU clock respectively.

Table 58. WDTREL reload value ($f_{\text{CPU}} = 40 \text{ MHz}$)

| Reload value in WDTREL | Prescaler for $f_{\text{CPU}} = 40 \text{ MHz}$ | |
|------------------------|---|---------------------|
| | 2 (WDTIN = '0') | 128 (WDTIN = '1') |
| FFh | 12.8 μs | 819.2 μs |
| 00h | 3.277ms | 209.7ms |

Table 59. WDTREL reload value ($f_{\text{CPU}} = 64 \text{ MHz}$)

| Reload value in WDTREL | Prescaler for $f_{\text{CPU}} = 64 \text{ MHz}$ | |
|------------------------|---|-------------------|
| | 2 (WDTIN = '0') | 128 (WDTIN = '1') |
| FFh | 8 μs | 512 μs |
| 00h | 2.048ms | 131.1ms |

19 System reset

System reset initializes the MCU in a predefined state. There are six ways to activate a reset state. The system start-up configuration is different for each case as shown in [Table 60](#).

Table 60. Reset event definition

| Reset Source | Flag | RPD Status | Conditions |
|----------------------------------|------|---------------|---|
| Power-on reset | PONR | Low | Power-on |
| Asynchronous Hardware reset | LHWR | Low | $t_{\overline{RSTIN}} > ^1)$ |
| Synchronous Long Hardware reset | | High | $t_{\overline{RSTIN}} > (1032 + 12) \text{ TCL} + \max(4 \text{ TCL}, 500\text{ns})$ |
| Synchronous Short Hardware reset | SHWR | High | $t_{\overline{RSTIN}} > \max(4 \text{ TCL}, 500\text{ns})$ $t_{\overline{RSTIN}} \leq (1032 + 12) \text{ TCL} + \max(4 \text{ TCL}, 500\text{ns})$ |
| Watchdog Timer reset | WDTR | ³⁾ | WDT overflow |
| Software reset | SWR | ³⁾ | SRST instruction execution |

¹⁾ \overline{RSTIN} pulse should be longer than 500ns (Filter) and than settling time for configuration of Port0.

²⁾ See next [Section 19.1](#) for more details on minimum reset pulse duration.

³⁾ The RPD status has no influence unless Bidirectional Reset is activated (bit `BDRSTEN` in `SYSCON`): RPD low inhibits the Bidirectional reset on SW and WDT reset events, that is \overline{RSTIN} is not activated (refer to [Sections 19.4](#), [19.5](#) and [19.6](#)).

19.1 Input filter

On \overline{RSTIN} input pin an on-chip RC filter is implemented. It is sized to filter all the spikes shorter than 50ns. On the other side, a valid pulse shall be longer than 500ns to grant that ST10 recognizes a reset command. In between 50ns and 500ns a pulse can either be filtered or recognized as valid, depending on the operating conditions and process variations.

For this reason all minimum durations mentioned in this Chapter for the different kind of reset events shall be carefully evaluated taking into account of the above requirements.

In particular, for Short Hardware Reset, where only 4 TCL is specified as minimum input reset pulse duration, the operating frequency is a key factor. Examples:

- For a CPU clock of 64 MHz, 4 TCL is 31.25ns, so it would be filtered. In this case the minimum becomes the one imposed by the filter (that is 500ns).
- For a CPU clock of 4 MHz, 4 TCL is 500ns. In this case the minimum from the formula is coherent with the limit imposed by the filter.

19.2 Asynchronous reset

An asynchronous reset is triggered when $\overline{\text{RSTIN}}$ pin is pulled low while RPD pin is at low level. Then the ST10F276 is immediately (after the input filter delay) forced in reset default state. It pulls low $\overline{\text{RSTOUT}}$ pin, it cancels pending internal hold states if any, it aborts all internal/external bus cycles, it switches buses (data, address and control signals) and I/O pin drivers to high-impedance, it pulls high Port0 pins.

Note: If an asynchronous reset occurs during a read or write phase in internal memories, the content of the memory itself could be corrupted: to avoid this, synchronous reset usage is strongly recommended.

Power-on reset

The asynchronous reset must be used during the power-on of the device. Depending on crystal or resonator frequency, the on-chip oscillator needs about 1ms to 10ms to stabilize (Refer to Electrical Characteristics Section), with an already stable V_{DD} . The logic of the ST10F276 does not need a stabilized clock signal to detect an asynchronous reset, so it is suitable for power-on conditions. To ensure a proper reset sequence, the $\overline{\text{RSTIN}}$ pin and the RPD pin must be held at low level until the device clock signal is stabilized and the system configuration value on Port0 is settled.

At Power-on it is important to respect some additional constraints introduced by the start-up phase of the different embedded modules.

In particular the on-chip voltage regulator needs at least 1ms to stabilize the internal 1.8V for the core logic: this time is computed from when the external reference (V_{DD}) becomes stable (inside specification range, that is at least 4.5V). This is a constraint for the application hardware (external voltage regulator): the $\overline{\text{RSTIN}}$ pin assertion shall be extended to guarantee the voltage regulator stabilization.

A second constraint is imposed by the embedded FLASH. When booting from internal memory, starting from $\overline{\text{RSTIN}}$ releasing, it needs a maximum of 1ms for its initialization: before that, the internal reset (RST signal) is not released, so the CPU does not start code execution in internal memory.

Note: This is not true if external memory is used (pin $\overline{\text{EA}}$ held low during reset phase). In this case, once $\overline{\text{RSTIN}}$ pin is released, and after few CPU clock (Filter delay plus 3...8 TCL), the internal reset signal RST is released as well, so the code execution can start immediately after. Obviously, an eventual access to the data in internal Flash is forbidden before its initialization phase is completed: an eventual access during starting phase will return FFFFh (just at the beginning), while later 009Bh (an illegal opcode trap can be generated).

At Power-on, the $\overline{\text{RSTIN}}$ pin shall be tied low for a minimum time that includes also the start-up time of the main oscillator ($t_{\text{STUP}} = 1\text{ms}$ for resonator, 10ms for crystal) and PLL synchronization time ($t_{\text{PSUP}} = 200\mu\text{s}$): this means that if the internal FLASH is used, the $\overline{\text{RSTIN}}$ pin could be released before the main oscillator and PLL are stable to recover some time in the start-up phase (FLASH initialization only needs stable V_{18} , but does not need stable system clock since an internal dedicated oscillator is used).

Warning: It is recommended to provide the external hardware with a current limitation circuitry. This is necessary to avoid permanent damages of the device during the power-on transient, when the capacitance on V_{18} pin is charged. For the on-chip voltage regulator functionality 10nF are

sufficient: anyway, a maximum of 100nF on V₁₈ pin should not generate problems of over-current (higher value is allowed if current is limited by the external hardware). External current limitation is anyway recommended also to avoid risks of damage in case of temporary short between V₁₈ and ground: the internal 1.8V drivers are sized to drive currents of several tens of Ampere, so the current shall be limited by the external hardware. The limit of current is imposed by power dissipation considerations (Refer to Electrical Characteristics Section).

In next Figures 24 and 25 Asynchronous Power-on timing diagrams are reported, respectively with boot from internal or external memory, highlighting the reset phase extension introduced by the embedded FLASH module when selected.

Note: Never power the device without keeping \overline{RSTIN} pin grounded: the device could enter in unpredictable states, risking also permanent damages.

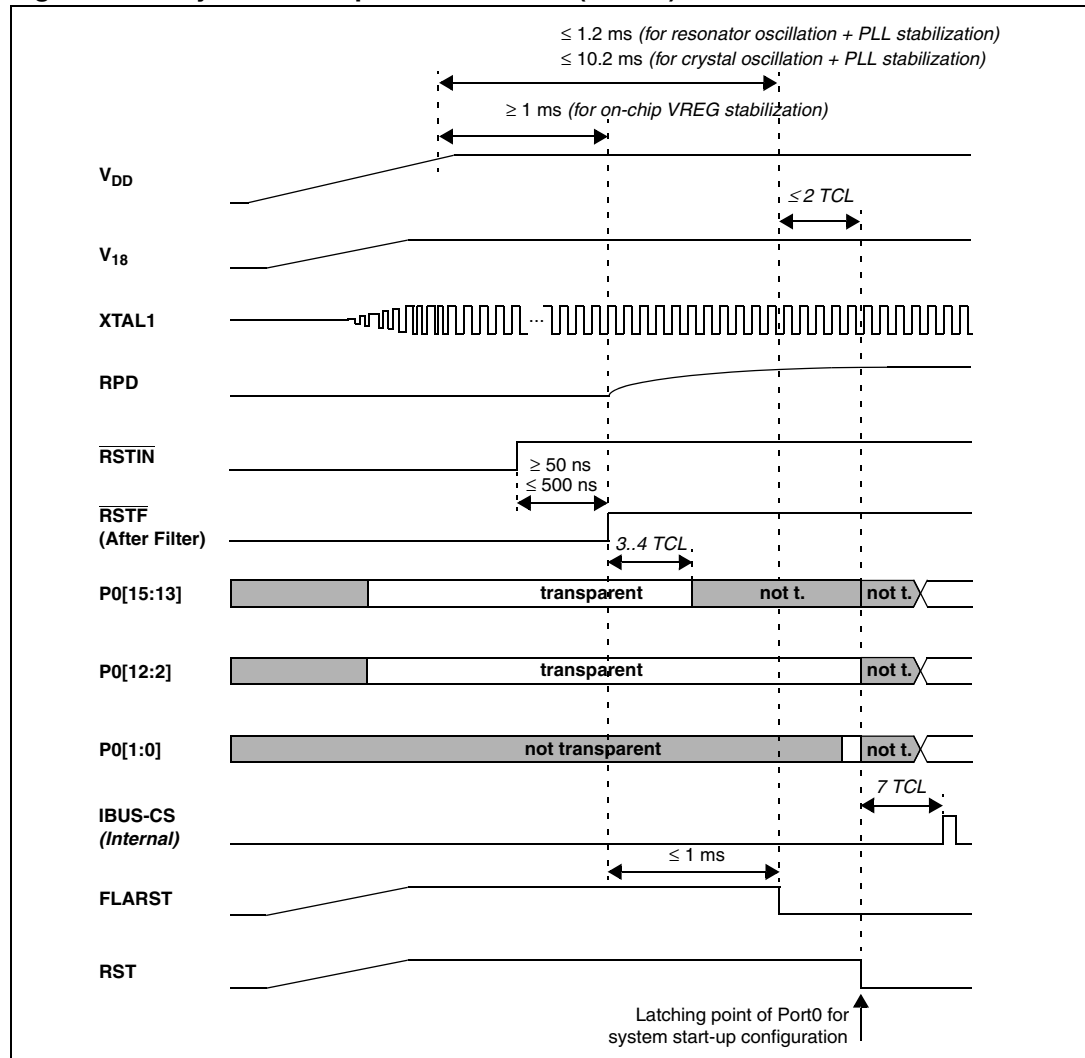
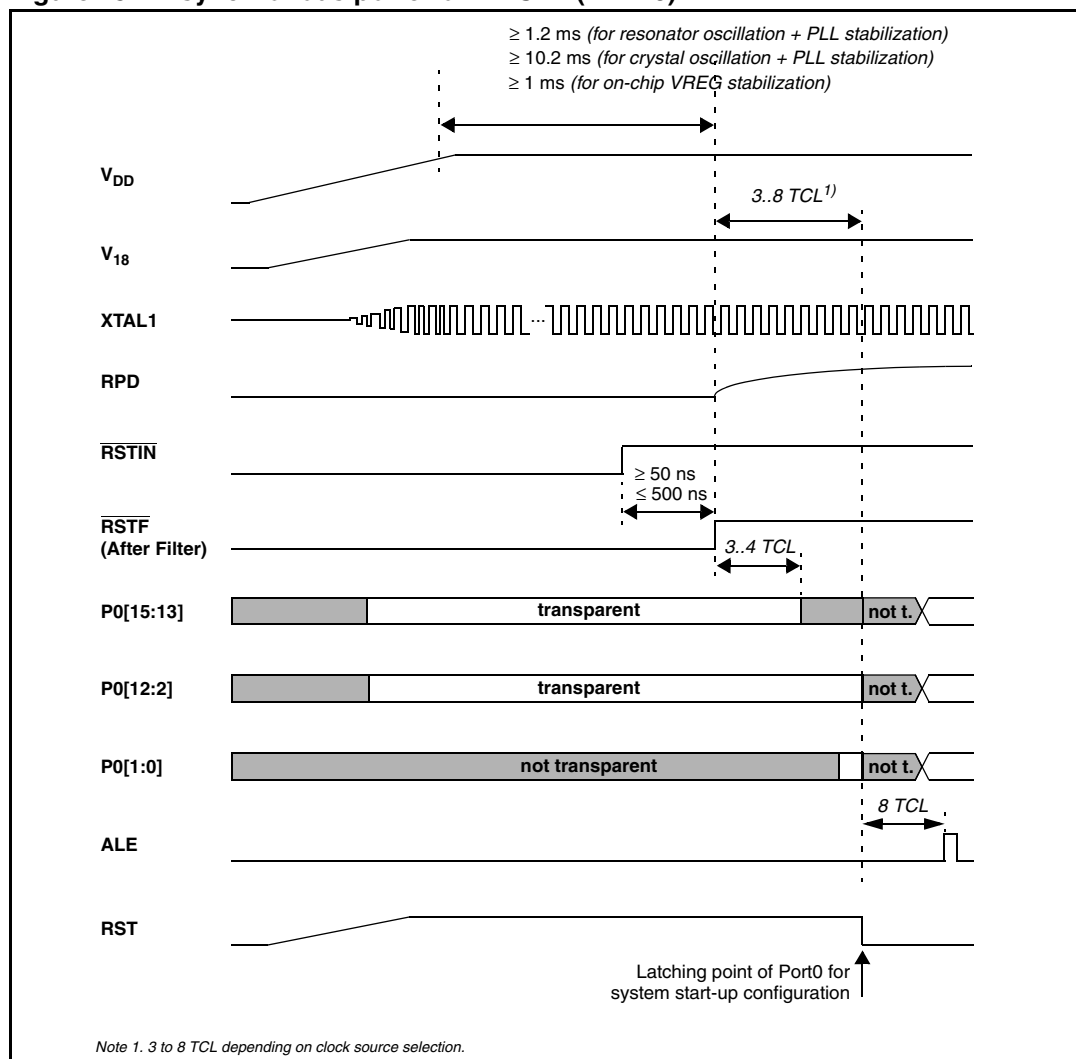
Figure 24. Asynchronous power-on RESET ($\overline{EA} = 1$)

Figure 25. Asynchronous power-on RESET ($\overline{EA} = 0$)

Hardware reset

The asynchronous reset must be used to recover from catastrophic situations of the application. It may be triggered by the hardware of the application. Internal hardware logic and application circuitry are described in Reset circuitry chapter and Figures 37, 38 and 39. It occurs when \overline{RSTIN} is low and RPD is detected (or becomes) low as well.

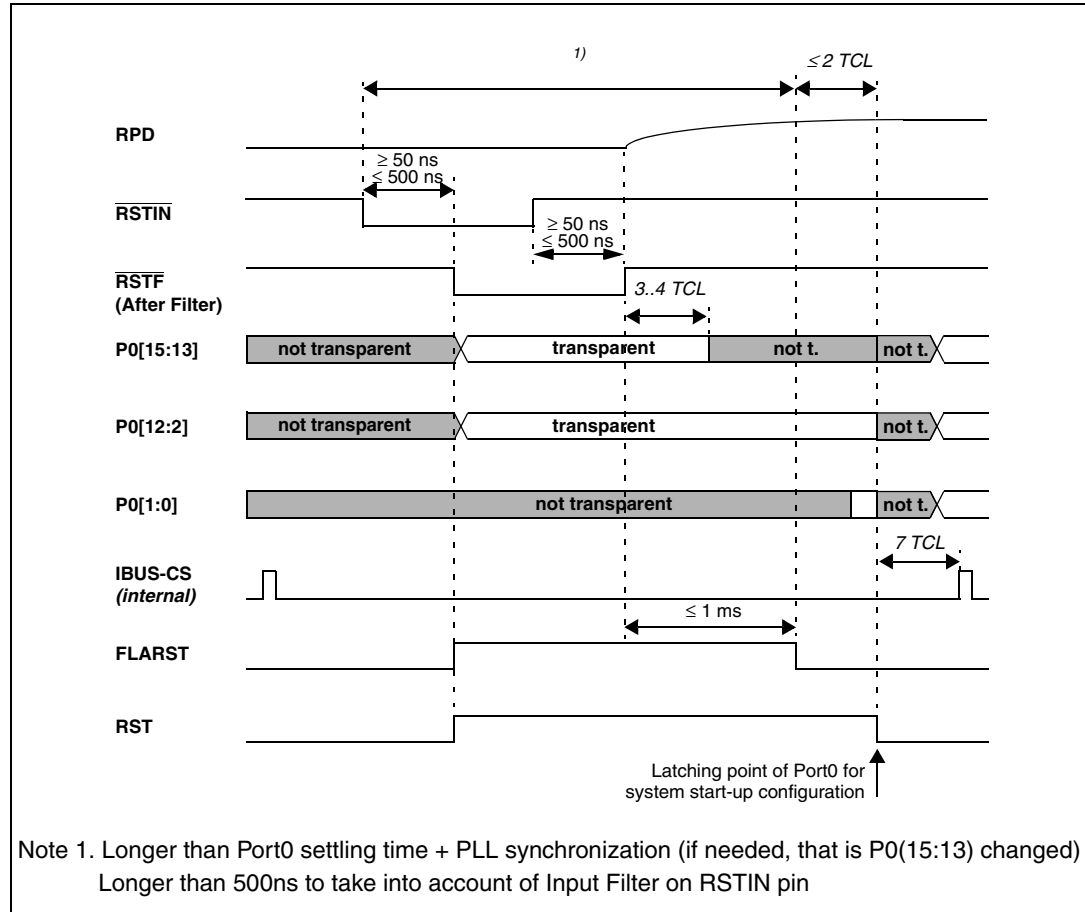
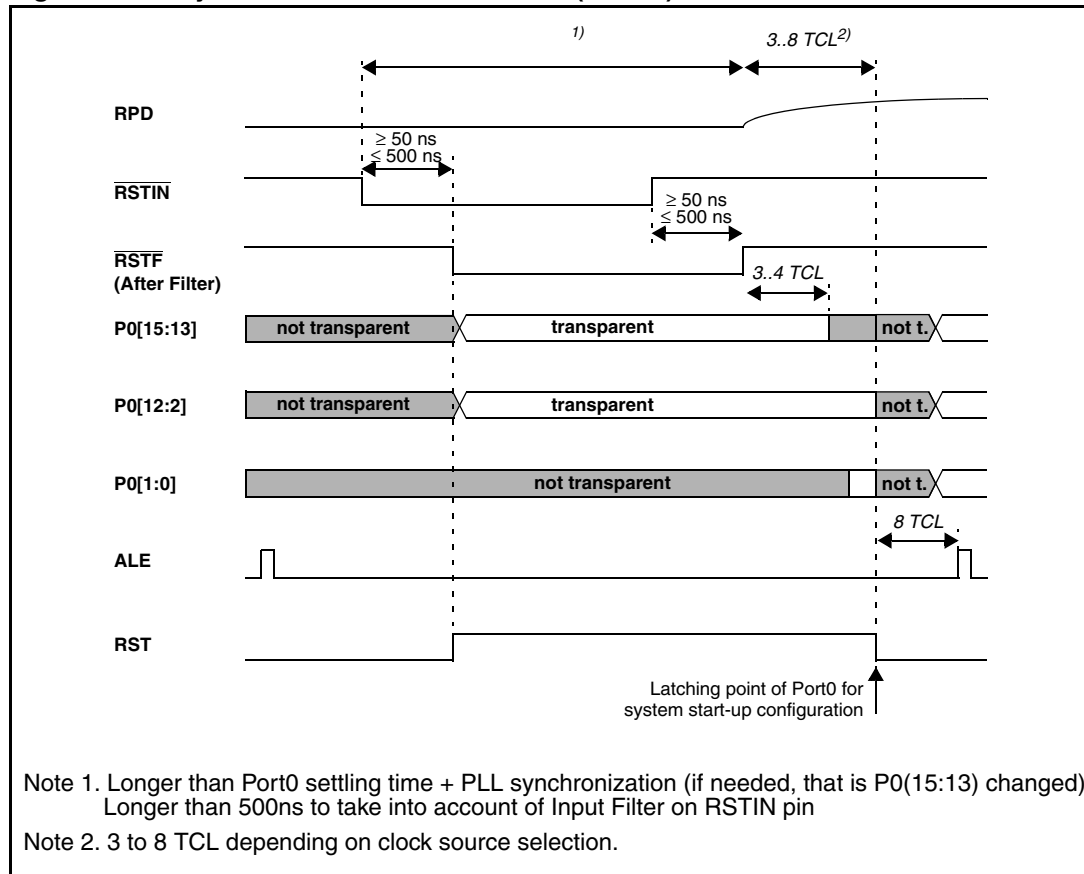
Figure 26. Asynchronous hardware RESET ($\overline{EA} = 1$)

Figure 27. Asynchronous hardware RESET ($\overline{EA} = 0$)**Exit from asynchronous reset state**

When the \overline{RSTIN} pin is pulled high, the device restarts: as already mentioned, if internal FLASH is used, the restarting occurs after the embedded FLASH initialization routine is completed. The system configuration is latched from Port0: ALE, \overline{RD} and $\overline{WR/WRL}$ pins are driven to their inactive level. The ST10F276 starts program execution from memory location 00'0000h in code segment 0. This starting location will typically point to the general initialization routine. Timing of asynchronous Hardware Reset sequence are summarized in [Figure 26](#) and [Figure 27](#).

19.3 Synchronous reset (warm reset)

A synchronous reset is triggered when \overline{RSTIN} pin is pulled low while RPD pin is at high level. In order to properly activate the internal reset logic of the device, the \overline{RSTIN} pin must be held low, at least, during 4 TCL (2 periods of CPU clock): refer also to [Section 19.1](#) for details on minimum reset pulse duration. The I/O pins are set to high impedance and \overline{RSTOUT} pin is driven low. After \overline{RSTIN} level is detected, a short duration of a maximum of 12 TCL (six periods of CPU clock) elapses, during which pending internal hold states are cancelled and the current internal access cycle if any is completed. External bus cycle is aborted. The internal pull-down of \overline{RSTIN} pin is activated if bit BDRSTEN of SYSCON register was previously set by software. Note that this bit is always cleared on power-on or after a reset sequence.

Short and long synchronous reset

Once the first maximum 16 TCL are elapsed (4+12TCL), the internal reset sequence starts. It is 1024 TCL cycles long: at the end of it, and after other 8TCL the level of $\overline{\text{RSTIN}}$ is sampled (after the filter, see $\overline{\text{RSTF}}$ in the drawings): if it is already at high level, only Short Reset is flagged (Refer to [Chapter 19: System reset](#) for details on reset flags); if it is recognized still low, the Long reset is flagged as well. The major difference between Long and Short reset is that during the Long reset, also P0(15:13) become transparent, so it is possible to change the clock options.

Warning: In case of a short pulse on $\overline{\text{RSTIN}}$ pin, and when Bidirectional reset is enabled, the $\overline{\text{RSTIN}}$ pin is held low by the internal circuitry. At the end of the 1024 TCL cycles, the $\overline{\text{RSTIN}}$ pin is released, but due to the presence of the input analog filter the internal input reset signal ($\overline{\text{RSTF}}$ in the drawings) is released later (from 50 to 500ns). This delay is in parallel with the additional 8 TCL, at the end of which the internal input reset line ($\overline{\text{RSTF}}$) is sampled, to decide if the reset event is Short or Long. In particular:

- If 8 TCL > 500ns ($F_{\text{CPU}} < 8 \text{ MHz}$), the reset event is always recognized as Short
- If 8 TCL < 500ns ($F_{\text{CPU}} > 8 \text{ MHz}$), the reset event could be recognized either as Short or Long, depending on the real filter delay (between 50 and 500ns) and the CPU frequency ($\overline{\text{RSTF}}$ sampled High means Short reset, $\overline{\text{RSTF}}$ sampled Low means Long reset). Note that in case a Long Reset is recognized, once the 8 TCL are elapsed, the P0(15:13) pins becomes transparent, so the system clock can be re-configured. The port returns not transparent 3-4TCL after the internal $\overline{\text{RSTF}}$ signal becomes high.

The same behavior just described, occurs also when unidirectional reset is selected and $\overline{\text{RSTIN}}$ pin is held low till the end of the internal sequence (exactly 1024TCL + max 16 TCL) and released exactly at that time.

Note: When running with CPU frequency lower than 40 MHz, the minimum valid reset pulse to be recognized by the CPU (4 TCL) could be longer than the minimum analog filter delay (50ns); so it might happen that a short reset pulse is not filtered by the analog input filter, but on the other hand it is not long enough to trigger a CPU reset (shorter than 4 TCL): this would generate a FLASH reset but not a system reset. In this condition, the FLASH answers always with FFFFh, which leads to an illegal opcode and consequently a trap event is generated.

Exit from synchronous reset state

The reset sequence is extended until $\overline{\text{RSTIN}}$ level becomes high. Besides, it is internally prolonged by the FLASH initialization when $\overline{\text{EA}}=1$ (internal memory selected). Then, the code execution restarts. The system configuration is latched from Port0, and ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}/\overline{\text{WRL}}$ pins are driven to their inactive level. The ST10F276 starts program execution from memory location 00'0000h in code segment 0. This starting location will typically point to the general initialization routine. Timing of synchronous reset sequence are summarized in Figures 28 and 29 where a Short Reset event is shown, with particular highlighting on the fact that it can degenerate into Long Reset: the two figures show the behavior when booting from internal or external memory respectively. Figures 30 and 31 reports the timing of a typical synchronous Long Reset, again when booting from internal or external memory.

Synchronous reset and RPD pin

Whenever the $\overline{\text{RSTIN}}$ pin is pulled low (by external hardware or as a consequence of a Bidirectional reset), the RPD internal weak pull-down is activated. The external capacitance (if any) on RPD pin is slowly discharged through the internal weak pull-down. If the voltage level on RPD pin reaches the input low threshold (around 2.5V), the reset event becomes immediately asynchronous. In case of hardware reset (short or long) the situation goes immediately to the one illustrated in [Figure 26](#). There is no effect if RPD comes again above the input threshold: the asynchronous reset is completed coherently. To grant the normal completion of a synchronous reset, the value of the capacitance shall be big enough to maintain the voltage on RPD pin sufficient high along the duration of the internal reset sequence.

For a Software or Watchdog reset events, an active synchronous reset is completed regardless of the RPD status.

It is important to highlight that the signal that makes RPD status transparent under reset is the internal $\overline{\text{RSTF}}$ (after the noise filter).

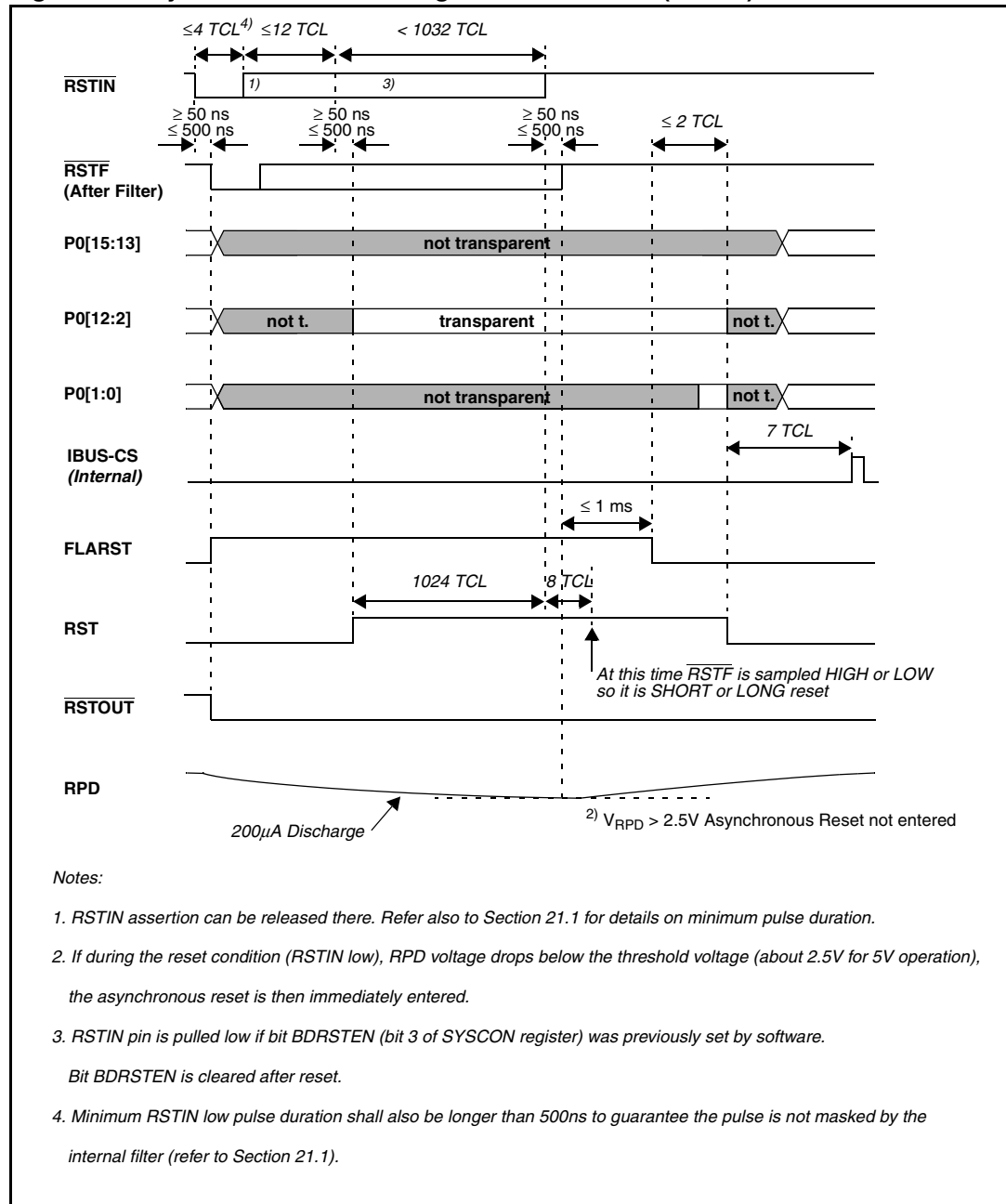
Figure 28. Synchronous short / long hardware RESET ($\overline{EA} = 1$)

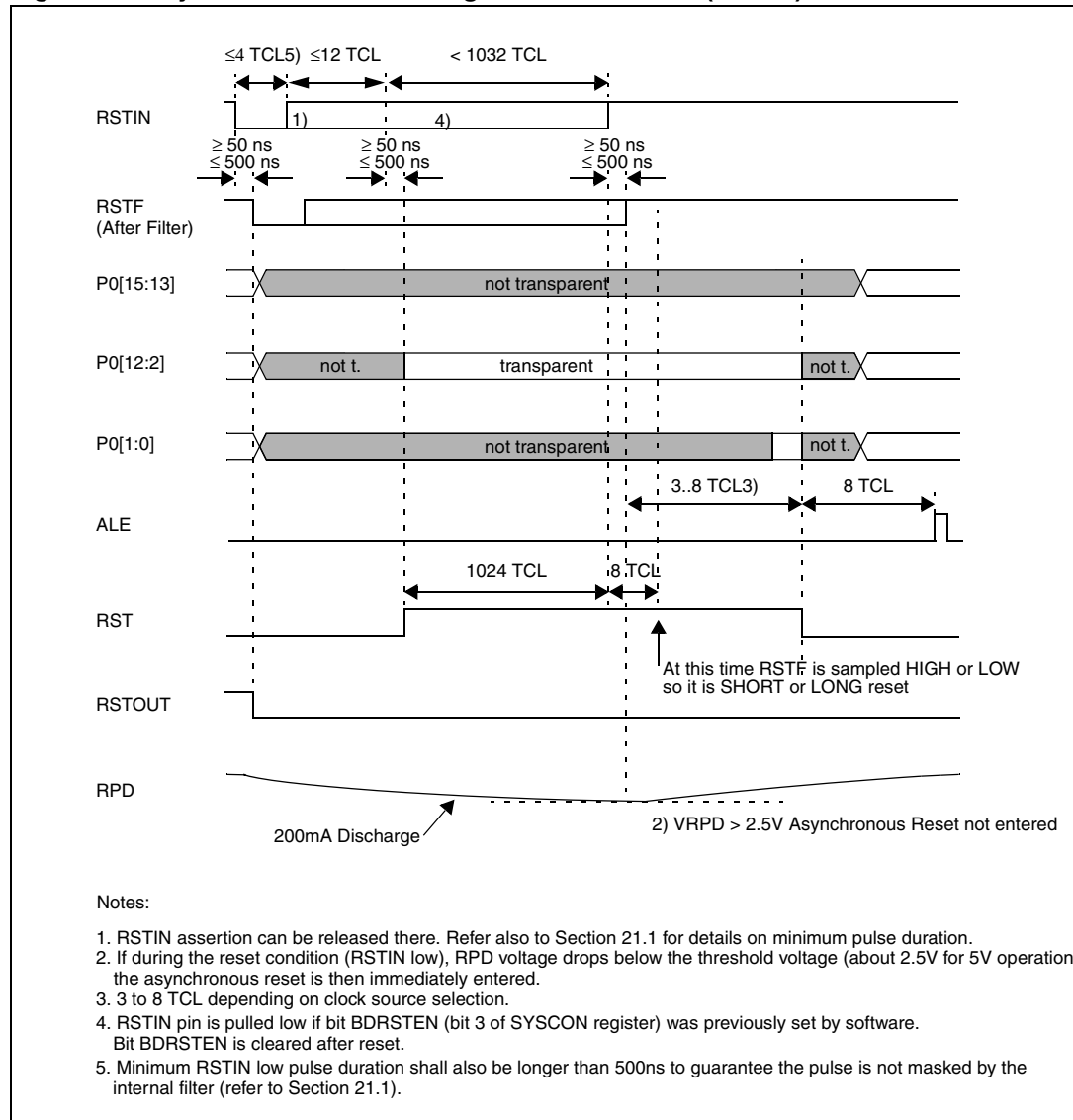
Figure 29. Synchronous short / long hardware RESET ($\overline{EA} = 0$)

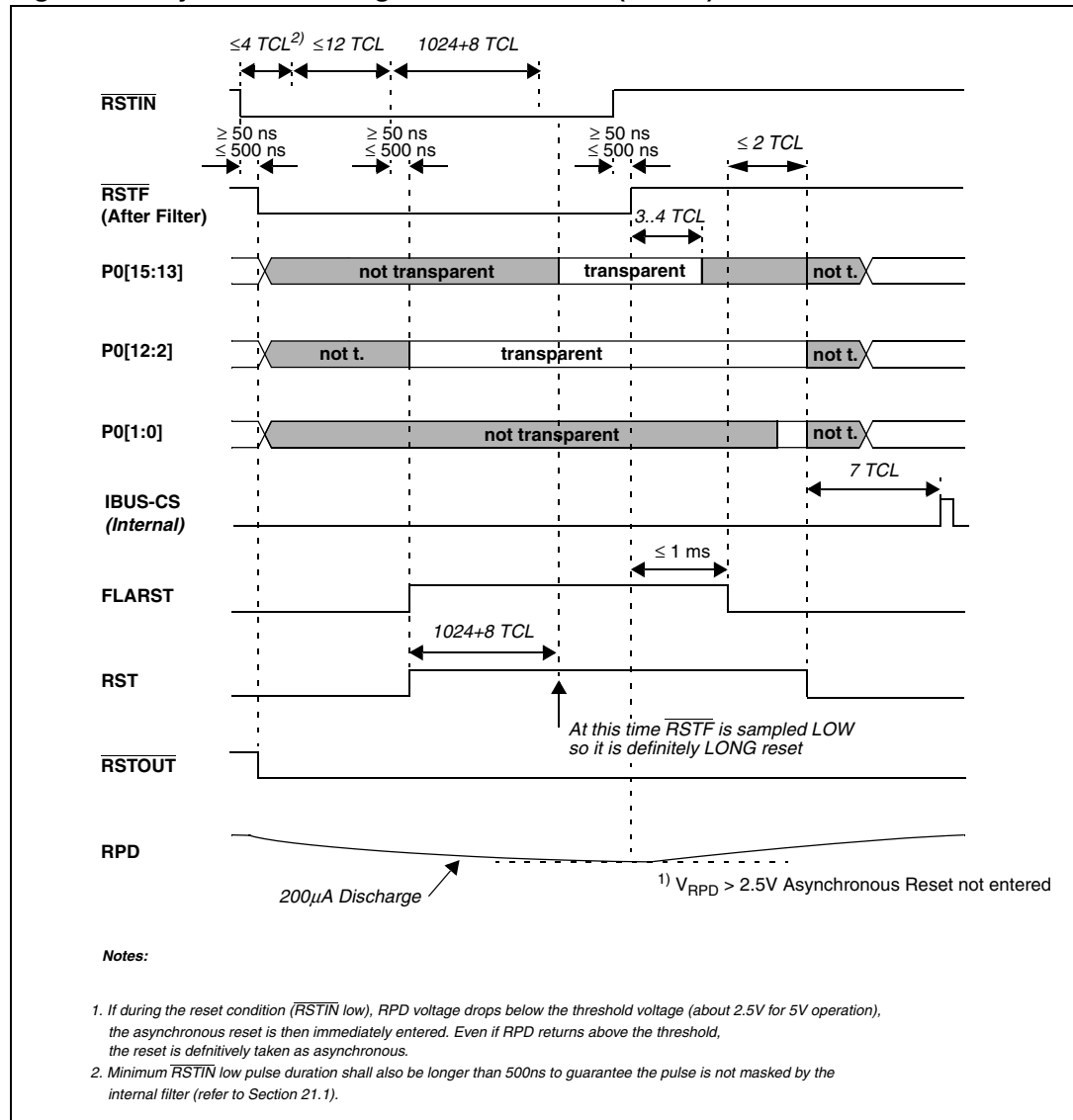
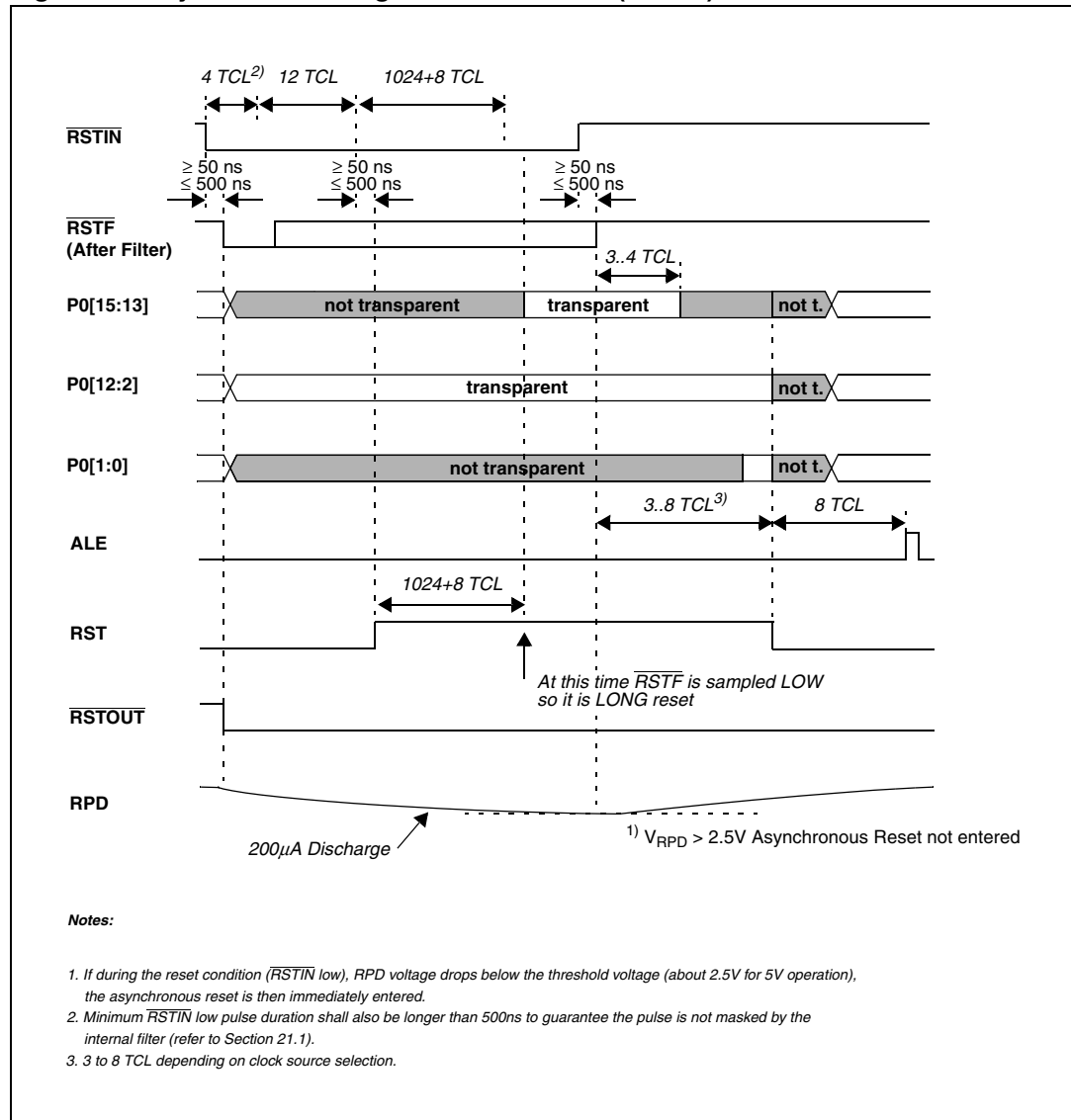
Figure 30. Synchronous long hardware RESET ($\overline{EA} = 1$)

Figure 31. Synchronous long hardware RESET ($\overline{EA} = 0$)

19.4 Software reset

A software reset sequence can be triggered at any time by the protected SRST (software reset) instruction. This instruction can be deliberately executed within a program, e.g. to leave bootstrap loader mode, or on a hardware trap that reveals system failure.

On execution of the SRST instruction, the internal reset sequence is started. The microcontroller behavior is the same as for a synchronous short reset, except that only bits P0.12...P0.8 are latched at the end of the reset sequence, while previously latched, bits P0.7...P0.2 are cleared (that is written at '1').

A Software reset is always taken as synchronous: there is no influence on Software Reset behavior with RPD status. In case Bidirectional Reset is selected, a Software Reset event pulls \overline{RSTIN} pin low: this occurs only if RPD is high; if RPD is low, \overline{RSTIN} pin is not pulled low even though Bidirectional Reset is selected.

Refer to next Figures 32 and 33 for unidirectional SW reset timing, and to Figures 34, 35 and 36 for bidirectional.

19.5 Watchdog timer reset

When the watchdog timer is not disabled during the initialization, or serviced regularly during program execution, it will overflow and trigger the reset sequence.

Unlike hardware and software resets, the watchdog reset completes a running external bus cycle if this bus cycle either does not use $\overline{\text{READY}}$, or if $\overline{\text{READY}}$ is sampled active (low) after the programmed wait states.

When $\overline{\text{READY}}$ is sampled inactive (high) after the programmed wait states the running external bus cycle is aborted. Then the internal reset sequence is started.

Bit P0.12...P0.8 are latched at the end of the reset sequence and bit P0.7...P0.2 are cleared (that is written at '1').

A Watchdog reset is always taken as synchronous: there is no influence on Watchdog Reset behavior with RPD status. In case Bidirectional Reset is selected, a Watchdog Reset event pulls $\overline{\text{RSTIN}}$ pin low: this occurs only if RPD is high; if RPD is low, $\overline{\text{RSTIN}}$ pin is not pulled low even though Bidirectional Reset is selected.

Refer to next Figures 32 and 33 for unidirectional SW reset timing, and to Figures 34, 35 and 36 for bidirectional.

Figure 32. SW / WDT unidirectional RESET ($\overline{\text{EA}} = 1$)

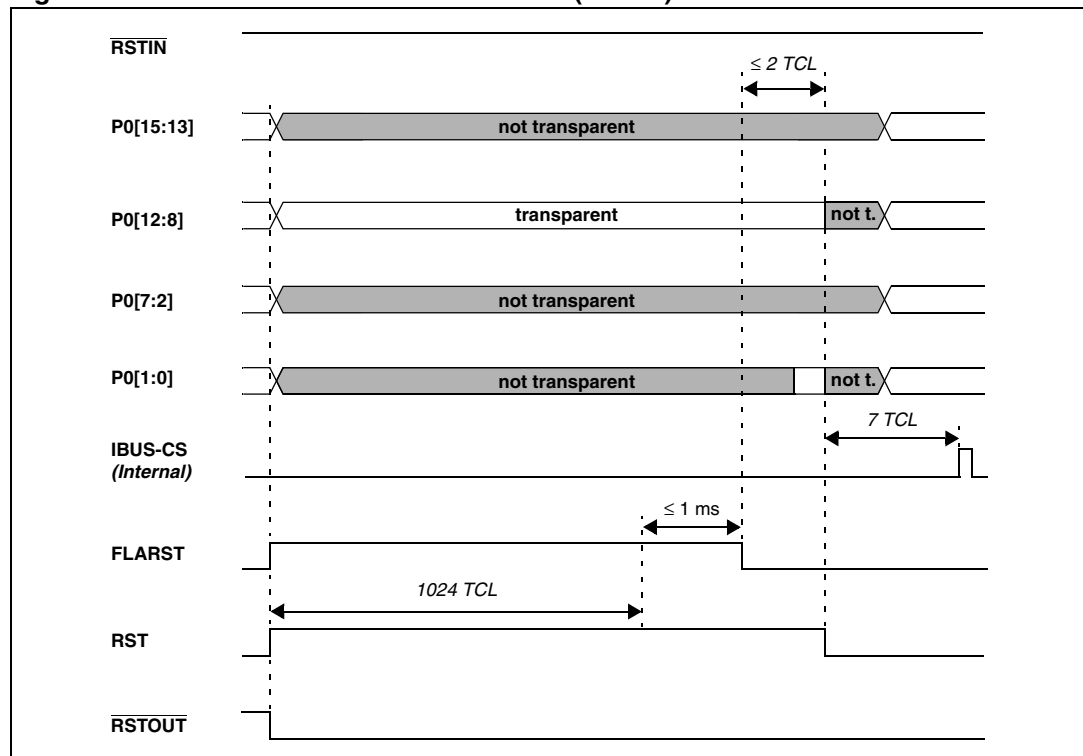
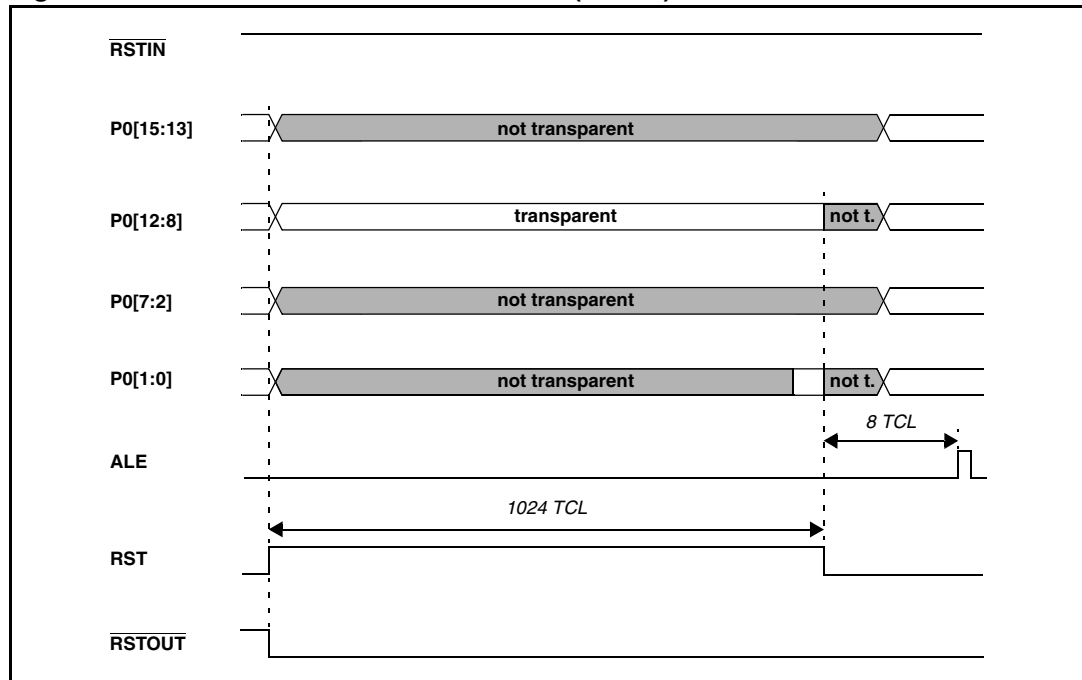


Figure 33. SW / WDT unidirectional RESET ($\overline{EA} = 0$)

19.6 Bidirectional reset

As shown in the previous sections, the \overline{RSTOUT} pin is driven active (low level) at the beginning of any reset sequence (synchronous/asynchronous hardware, software and watchdog timer resets). \overline{RSTOUT} pin stays active low beyond the end of the initialization routine, until the protected EINIT instruction (End of Initialization) is completed.

The Bidirectional Reset function is useful when external devices require a reset signal but cannot be connected to \overline{RSTOUT} pin, because \overline{RSTOUT} signal lasts during initialization. It is, for instance, the case of external memory running initialization routine before the execution of EINIT instruction.

Bidirectional reset function is enabled by setting bit 3 (BDRSTEN) in SYSCON register. It only can be enabled during the initialization routine, before EINIT instruction is completed.

When enabled, the open drain of the \overline{RSTIN} pin is activated, pulling down the reset signal, for the duration of the internal reset sequence (synchronous/asynchronous hardware, synchronous software and synchronous watchdog timer resets). At the end of the internal reset sequence the pull down is released and:

- After a Short Synchronous Bidirectional Hardware Reset, if \overline{RSTF} is sampled low 8 TCL periods after the internal reset sequence completion (refer to [Figure 28](#) and [Figure 29](#)), the Short Reset becomes a Long Reset. On the contrary, if \overline{RSTF} is sampled high the device simply exits reset state.
- After a Software or Watchdog Bidirectional Reset, the device exits from reset. If \overline{RSTF} remains still low for at least 4 TCL periods (minimum time to recognize a Short Hardware reset) after the reset exiting (refer to [Figure 34](#) and [Figure 35](#)), the Software or Watchdog Reset become a Short Hardware Reset. On the contrary, if \overline{RSTF} remains low for less than 4 TCL, the device simply exits reset state.

The Bidirectional reset is not effective in case RPD is held low, when a Software or Watchdog reset event occurs. On the contrary, if a Software or Watchdog Bidirectional reset event is active and RPD becomes low, the $\overline{\text{RSTIN}}$ pin is immediately released, while the internal reset sequence is completed regardless of RPD status change (1024 TCL).

Note: The bidirectional reset function is disabled by any reset sequence (bit BDRSTEN of SYSCON is cleared). To be activated again it must be enabled during the initialization routine.

WDTCN flags

Similarly to what already highlighted in the previous section when discussing about Short reset and the degeneration into Long reset, similar situations may occur when Bidirectional reset is enabled. The presence of the internal filter on $\overline{\text{RSTIN}}$ pin introduces a delay: when $\overline{\text{RSTIN}}$ is released, the internal signal after the filter (see $\overline{\text{RSTF}}$ in the drawings) is delayed, so it remains still active (low) for a while. It means that depending on the internal clock speed, a short reset may be recognized as a long reset: the WDTCN flags are set accordingly.

Besides, when either Software or Watchdog bidirectional reset events occur, again when the $\overline{\text{RSTIN}}$ pin is released (at the end of the internal reset sequence), the $\overline{\text{RSTF}}$ internal signal (after the filter) remains low for a while, and depending on the clock frequency it is recognized high or low: 8TCL after the completion of the internal sequence, the level of $\overline{\text{RSTF}}$ signal is sampled, and if recognized still low a Hardware reset sequence starts, and WDTCN will flag this last event, masking the previous one (Software or Watchdog reset). Typically, a Short Hardware reset is recognized, unless the $\overline{\text{RSTIN}}$ pin (and consequently internal signal $\overline{\text{RSTF}}$) is sufficiently held low by the external hardware to inject a Long Hardware reset. After this occurrence, the initialization routine is not able to recognize a Software or Watchdog bidirectional reset event, since a different source is flagged inside WDTCN register. This phenomenon does not occur when internal FLASH is selected during reset ($\overline{\text{EA}} = 1$), since the initialization of the FLASH itself extend the internal reset duration well beyond the filter delay.

Next Figures 34, 35 and 36 summarize the timing for Software and Watchdog Timer Bidirectional reset events: In particular Figure 36 shows the degeneration into Hardware reset.

Figure 34. SW / WDT bidirectional RESET ($\overline{EA}=1$)

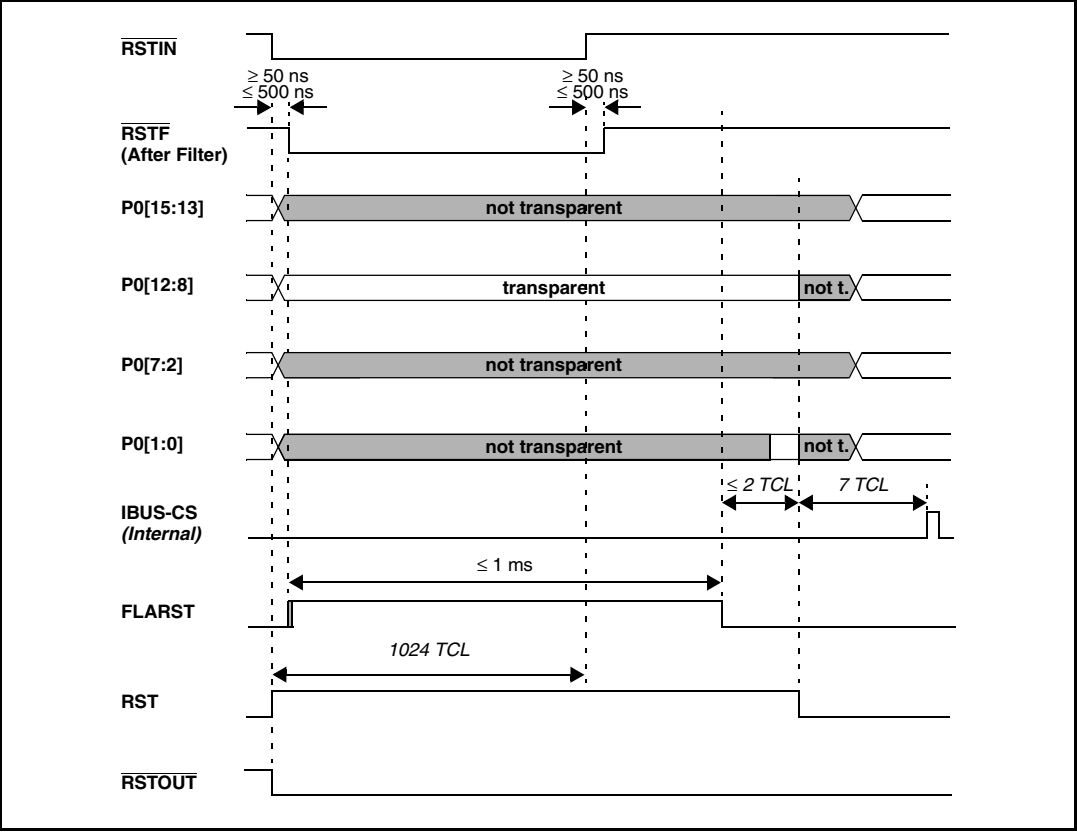


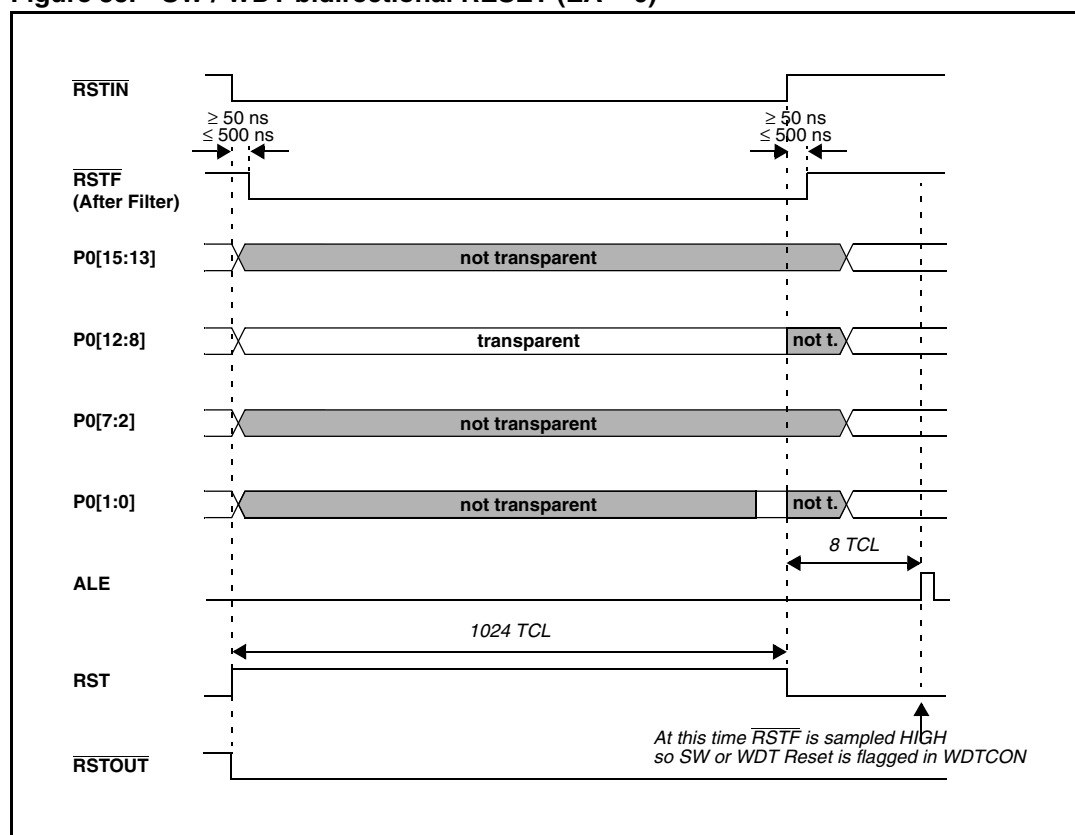
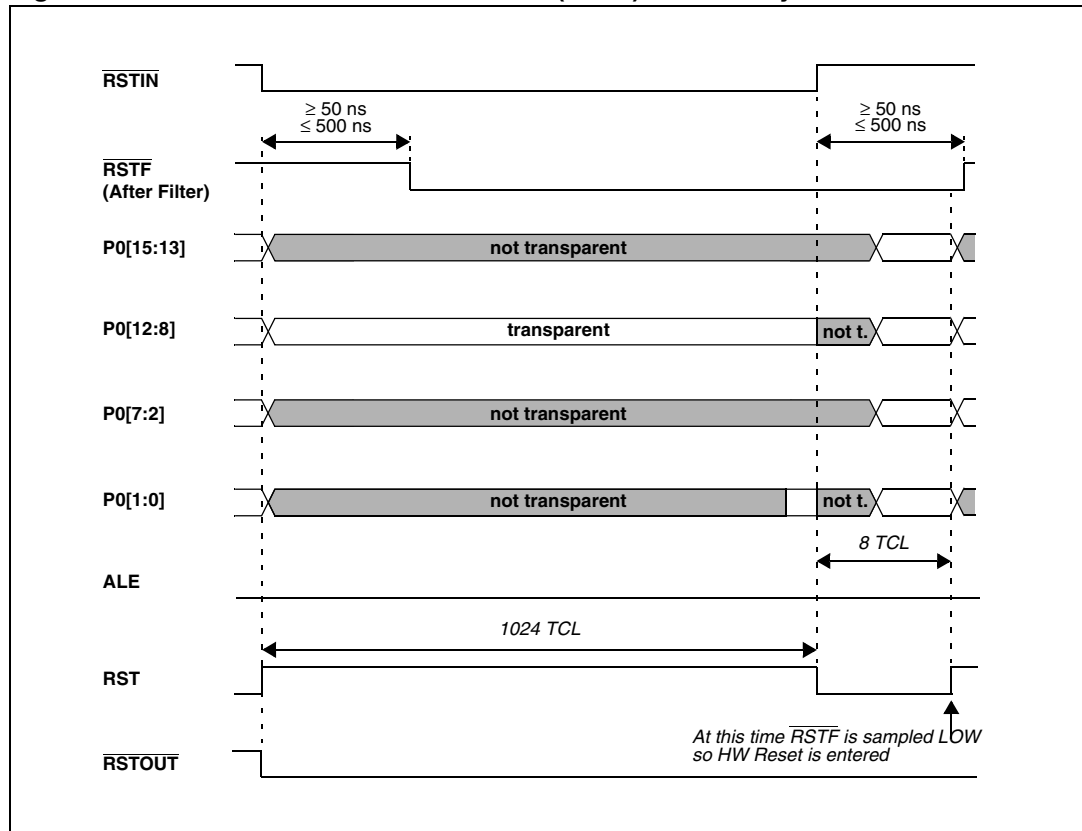
Figure 35. SW / WDT bidirectional RESET ($\overline{EA} = 0$)

Figure 36. SW / WDT bidirectional RESET ($\overline{EA}=0$) followed by a HW RESET

19.7 Reset circuitry

Internal reset circuitry is described in [Figure 39](#). The \overline{RSTIN} pin provides an internal pull-up resistor of $50\text{k}\Omega$ to $250\text{k}\Omega$ (The minimum reset time must be calculated using the lowest value).

It also provides a programmable (BDRSTEN bit of SYSCON register) pull-down to output internal reset state signal (synchronous reset, watchdog timer reset or software reset).

This bidirectional reset function is useful in applications where external devices require a reset signal but cannot be connected to \overline{RSTOUT} pin.

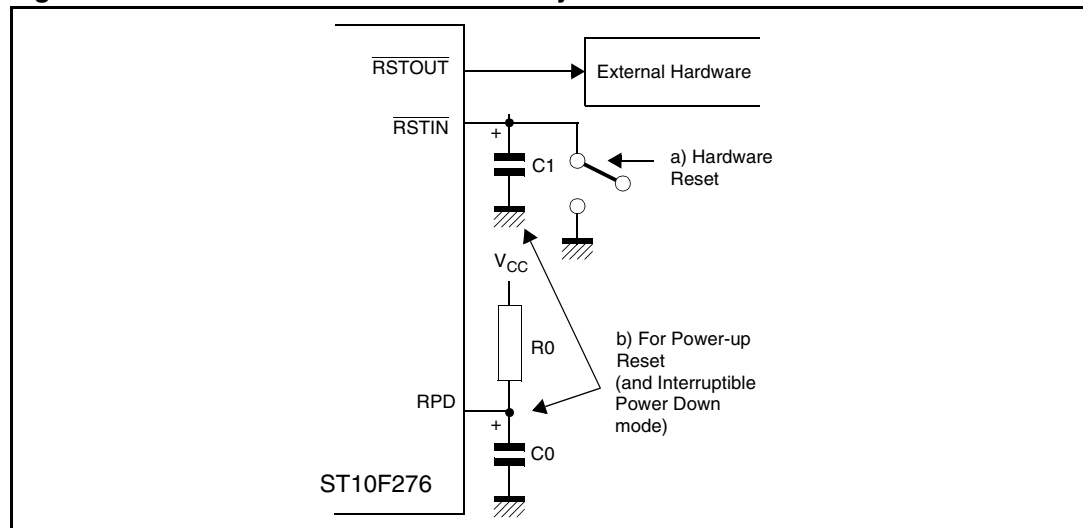
This is the case of an external memory running codes before EINIT (end of initialization) instruction is executed. \overline{RSTOUT} pin is pulled high only when EINIT is executed.

The RPD pin provides an internal weak pull-down resistor which discharges external capacitor at a typical rate of $200\mu\text{A}$. If bit PWDCFG of SYSCON register is set, an internal pull-up resistor is activated at the end of the reset sequence. This pull-up will charge any capacitor connected on RPD pin.

The simplest way to reset the ST10F276 is to insert a capacitor C1 between \overline{RSTIN} pin and V_{SS} , and a capacitor between RPD pin and V_{SS} (C0) with a pull-up resistor R0 between RPD pin and V_{DD} . The input \overline{RSTIN} provides an internal pull-up device equalling a resistor of $50\text{k}\Omega$ to $250\text{k}\Omega$ (the minimum reset time must be determined by the lowest value). Select C1 that produce a sufficient discharge time to permit the internal or external oscillator and / or internal PLL and the on-chip voltage regulator to stabilize.

To ensure correct power-up reset with controlled supply current consumption, specially if clock signal requires a long period of time to stabilize, an asynchronous hardware reset is required during power-up. For this reason, it is recommended to connect the external R0-C0 circuit shown in Figure 37 to the RPD pin. On power-up, the logical low level on RPD pin forces an asynchronous hardware reset when $\overline{\text{RSTIN}}$ is asserted low. The external pull-up R0 will then charge the capacitor C0. Note that an internal pull-down device on RPD pin is turned on when $\overline{\text{RSTIN}}$ pin is low, and causes the external capacitor (C0) to begin discharging at a typical rate of 100-200 μA . With this mechanism, after power-up reset, short low pulses applied on $\overline{\text{RSTIN}}$ produce synchronous hardware reset. If $\overline{\text{RSTIN}}$ is asserted longer than the time needed for C0 to be discharged by the internal pull-down device, then the device is forced in an asynchronous reset. This mechanism insures recovery from very catastrophic failure.

Figure 37. Minimum external reset circuitry



The minimum reset circuit of [Figure 37](#) is not adequate when the $\overline{\text{RSTIN}}$ pin is driven from the ST10F276 itself during software or watchdog triggered resets, because of the capacitor C1 that will keep the voltage on $\overline{\text{RSTIN}}$ pin above V_{IL} after the end of the internal reset sequence, and thus will trigger an asynchronous reset sequence.

Figure 38 shows an example of a reset circuit. In this example, R1-C1 external circuit is only used to generate power-up or manual reset, and R0-C0 circuit on RPD is used for power-up reset and to exit from Power Down mode. Diode D1 creates a wired-OR gate connection to the reset pin and may be replaced by open-collector Schmitt trigger buffer. Diode D2 provides a faster cycle time for repetitive power-on resets.

R2 is an optional pull-up for faster recovery and correct biasing of TTL Open Collector drivers.

Figure 38. System reset circuit

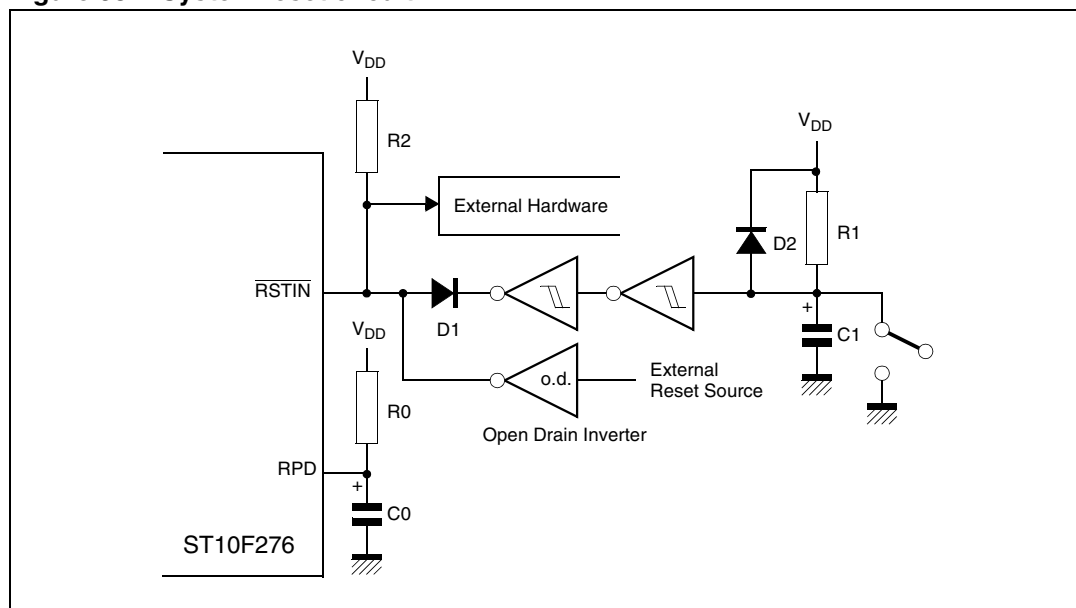


Figure 39. Internal (simplified) reset circuitry

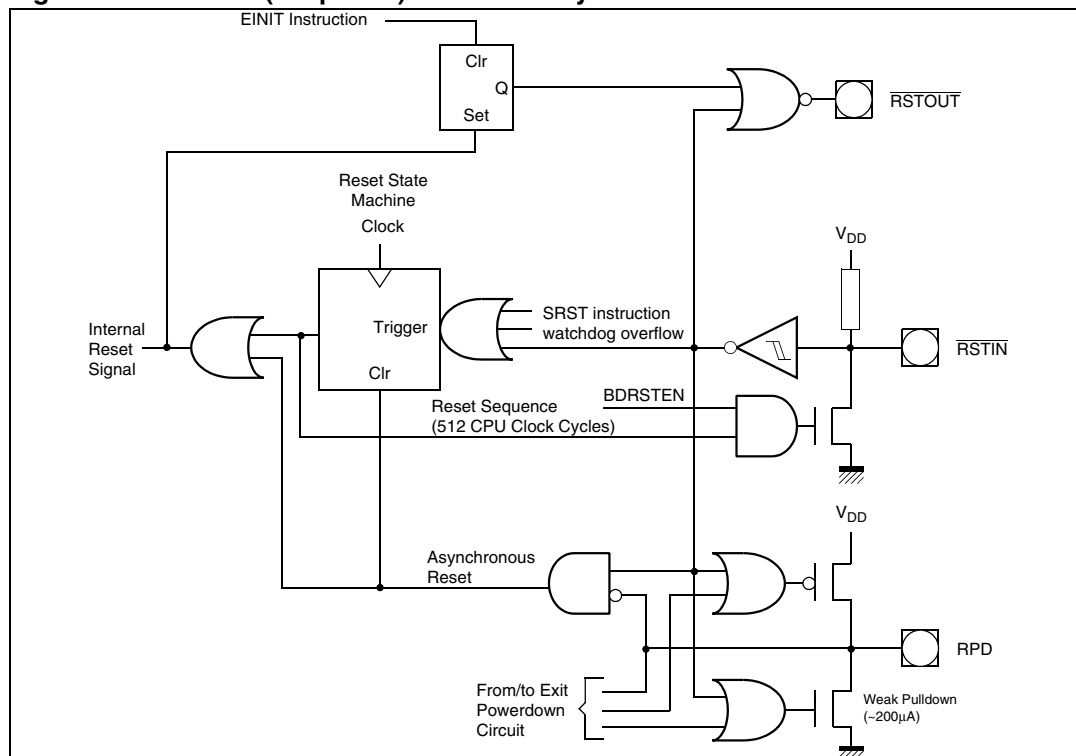
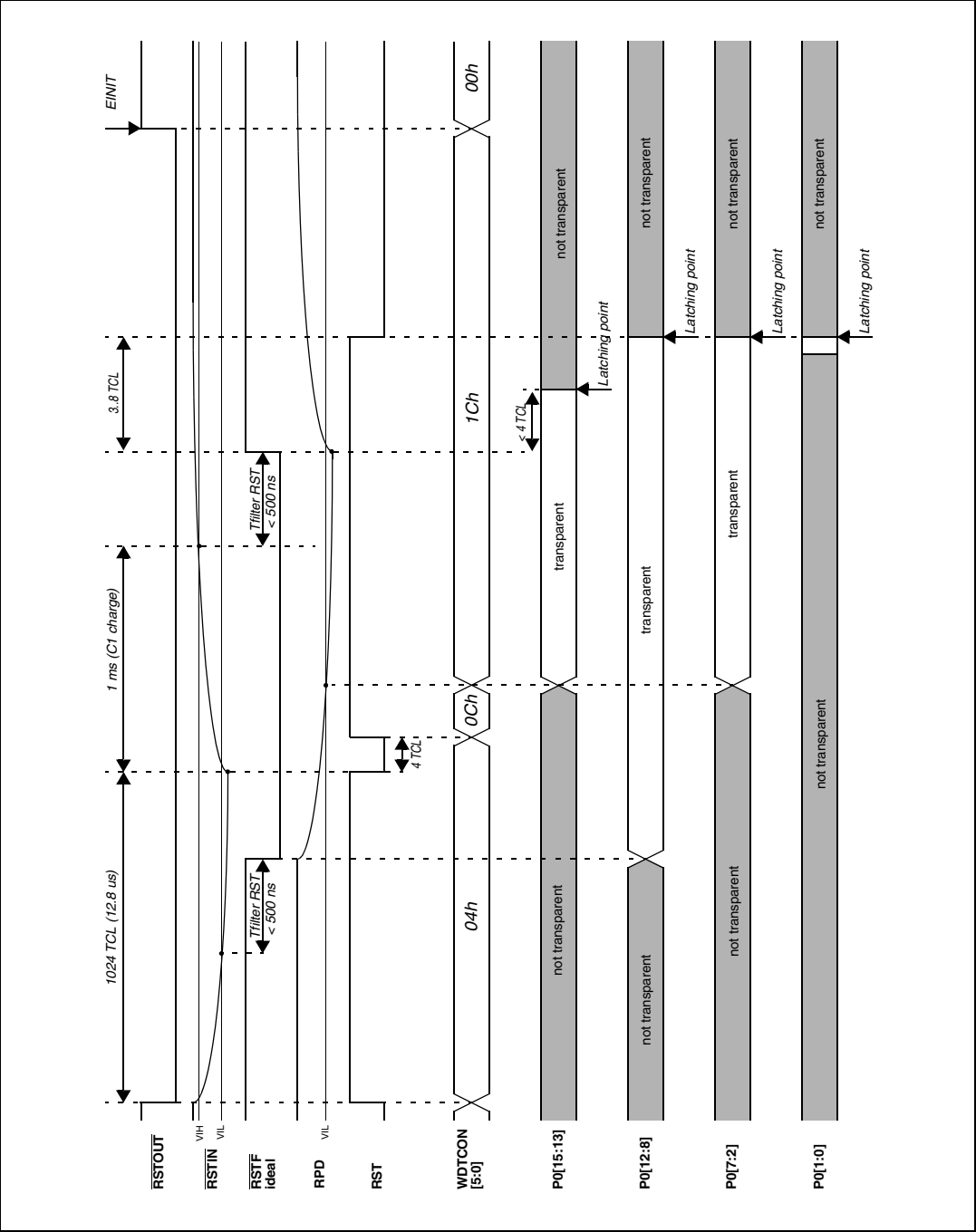


Figure 41. Example of software or watchdog bidirectional reset ($\overline{EA} = 0$)



19.9 Reset summary

A summary of the different reset events is reported in the table below.

Table 61. Reset event

| Event | RPD | EA | Bidir | Synch. Asynch. | RSTIN | | WDTCN Flags | | | | |
|---|-----|----|-------|-------------------|---|--------------------------------------|-------------|------|------|-----|------|
| | | | | | min | max | PONR | LHWR | SHWR | SWR | WDTR |
| Power-on Reset | 0 | 0 | N | Asynch. | 1 ms (VREG) 1.2 ms (Reson. + PLL) 10.2 ms (Crystal + PLL) | - | 1 | 1 | 1 | 1 | 0 |
| | 0 | 1 | N | Asynch. | 1ms (VREG) | - | 1 | 1 | 1 | 1 | 0 |
| | 1 | x | x | FORBIDDEN | | | | | | | |
| | x | x | Y | NOT APPLICABLE | | | | | | | |
| Hardware Reset (Asynchronous) | 0 | 0 | N | Asynch. | 500ns | - | 0 | 1 | 1 | 1 | 0 |
| | 0 | 1 | N | Asynch. | 500ns | - | 0 | 1 | 1 | 1 | 0 |
| | 0 | 0 | Y | Asynch. | 500ns | - | 0 | 1 | 1 | 1 | 0 |
| | 0 | 1 | Y | Asynch. | 500ns | - | 0 | 1 | 1 | 1 | 0 |
| Short Hardware Reset (Synchronous) ⁽¹⁾ | 1 | 0 | N | Synch. | max (4 TCL, 500ns) | 1032 + 12 TCL + max(4 TCL, 500ns) | 0 | 0 | 1 | 1 | 0 |
| | 1 | 1 | N | Synch. | max (4 TCL, 500ns) | 1032 + 12 TCL + max(4 TCL, 500ns) | 0 | 0 | 1 | 1 | 0 |
| | 1 | 0 | Y | Synch. | max (4 TCL, 500ns) | 1032 + 12 TCL + max(4 TCL, 500ns) | 0 | 0 | 1 | 1 | 0 |
| | | | | | Activated by internal logic for 1024 TCL | | | | | | |
| | 1 | 1 | Y | Synch. | max (4 TCL, 500ns) | 1032 + 12 TCL + max(4 TCL, 500ns) | 0 | 0 | 1 | 1 | 0 |
| | | | | | Activated by internal logic for 1024 TCL | | | | | | |
| Long Hardware Reset (Synchronous) | 1 | 0 | N | Synch. | 1032 + 12 TCL + max(4 TCL, 500ns) | - | 0 | 1 | 1 | 1 | 0 |
| | 1 | 1 | N | Synch. | 1032 + 12 TCL + max(4 TCL, 500ns) | - | 0 | 1 | 1 | 1 | 0 |
| | 1 | 0 | Y | Synch. | 1032 + 12 TCL + max(4 TCL, 500ns) | - | 0 | 1 | 1 | 1 | 0 |
| | | | | | Activated by internal logic only for 1024 TCL | | | | | | |
| | 1 | 1 | Y | Synch. | 1032 + 12 TCL + max(4 TCL, 500ns) | - | 0 | 1 | 1 | 1 | 0 |
| | | | | | Activated by internal logic only for 1024 TCL | | | | | | |

Table 61. Reset event (continued)

| Event | RPD | EA | Bidir | Synch. Asynch. | RSTIN | | WDTCN Flags | | | | |
|-------------------------------|-----|----|-------|-------------------|--|-----|-------------|------|------|-----|------|
| | | | | | min | max | PONR | LHWR | SHWR | SWR | WDTR |
| Software Reset ⁽²⁾ | x | 0 | N | Synch. | Not activated | | 0 | 0 | 0 | 1 | 0 |
| | x | 0 | N | Synch. | Not activated | | 0 | 0 | 0 | 1 | 0 |
| | 0 | 1 | Y | Synch. | Not activated | | 0 | 0 | 0 | 1 | 0 |
| | 1 | 1 | Y | Synch. | Activated by internal logic for 1024 TCL | | 0 | 0 | 0 | 1 | 0 |
| Watchdog Reset ⁽²⁾ | x | 0 | N | Synch. | Not activated | | 0 | 0 | 0 | 1 | 1 |
| | x | 0 | N | Synch. | Not activated | | 0 | 0 | 0 | 1 | 1 |
| | 0 | 1 | Y | Synch. | Not activated | | 0 | 0 | 0 | 1 | 1 |
| | 1 | 1 | Y | Synch. | Activated by internal logic for 1024 TCL | | 0 | 0 | 0 | 1 | 1 |

1. It can degenerate into a Long Hardware Reset and consequently differently flagged (see [Section 19.3](#) for details).
2. When Bidirectional is active (and with RPD=0), it can be followed by a Short Hardware Reset and consequently differently flagged (see [Section 19.6](#) for details).

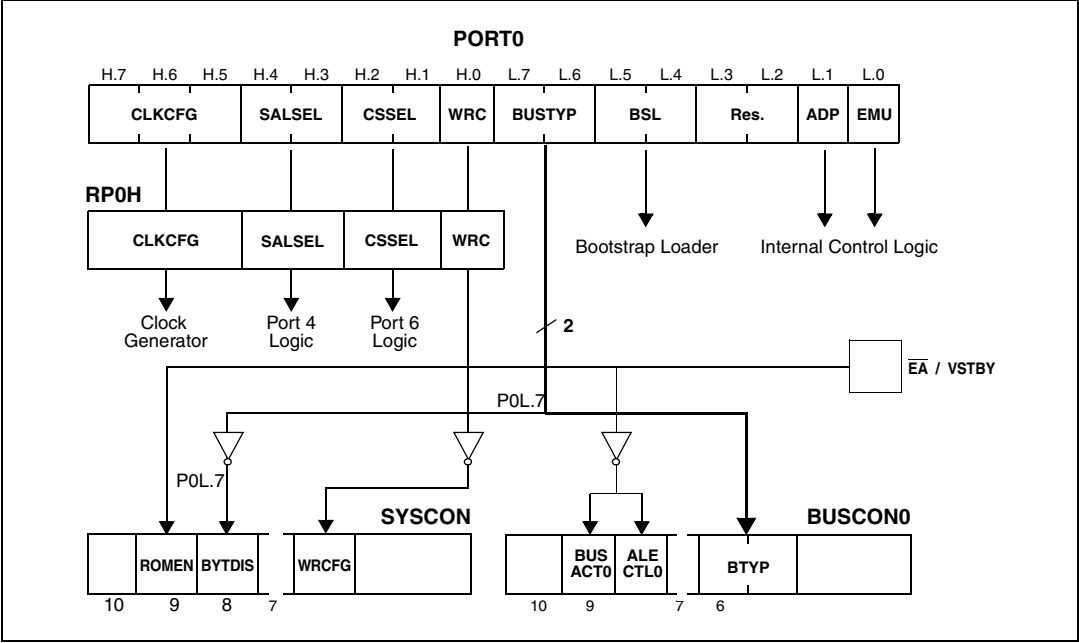
The start-up configurations and some system features are selected on reset sequences as described in [Table 62](#) and [Figure 42](#).

[Table 62](#) describes what is the system configuration latched on PORT0 in the six different reset modes. [Figure 42](#) summarizes the state of bits of PORT0 latched in RP0H, SYSCON, BUSCON0 registers.

Table 62. PORT0 latched configuration for the different reset events

| X: Pin is sampled -: Pin is not sampled | PORT0 | | | | | | | | | | | | | | | |
|--|---------------|-------|-------|-------------------|-------|--------------|-------|------------|----------|-------|----------|-------|----------|----------|------------|----------|
| | Clock Options | | | Segm. Addr. Lines | | Chip Selects | | WR config. | Bus Type | | Reserved | BSL | Reserved | Reserved | Adapt Mode | Emu Mode |
| | P0H.7 | P0H.6 | P0H.5 | P0H.4 | P0H.3 | P0H.2 | P0H.1 | P0H.0 | P0L.7 | P0L.6 | P0L.5 | P0L.4 | P0L.3 | P0L.2 | P0L.1 | P0L.0 |
| Sample event | | | | | | | | | | | | | | | | |
| Software Reset | - | - | - | X | X | X | X | X | X | X | - | - | - | - | - | - |
| Watchdog Reset | - | - | - | X | X | X | X | X | X | X | - | - | - | - | - | - |
| Synchronous Short Hardware Reset | - | - | - | X | X | X | X | X | X | X | X | X | X | X | X | X |
| Synchronous Long Hardware Reset | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| Asynchronous Hardware Reset | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |
| Asynchronous Power-On Reset | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X | X |

Figure 42. PORT0 bits latched into the different registers after reset



20 Power reduction modes

Three different power reduction modes with different levels of power reduction have been implemented in the ST10F276. In Idle mode only CPU is stopped, while peripheral still operate. In Power Down mode both CPU and peripherals are stopped. In Stand-by mode the main power supply (V_{DD}) can be turned off while a portion of the internal RAM remains powered via V_{STBY} dedicated power pin.

Idle and Power Down modes are software activated by a protected instruction and are terminated in different ways as described in the following sections.

Stand-by mode is entered simply removing V_{DD} , holding the MCU under reset state.

*Note: All external bus actions are completed before Idle or Power Down mode is entered. However, Idle or Power Down mode is **not** entered if READY is enabled, but has not been activated (driven low for negative polarity, or driven high for positive polarity) during the last bus access.*

20.1 Idle mode

Idle mode is entered by running IDLE protected instruction. The CPU operation is stopped and the peripherals still run.

Idle mode is terminate by any interrupt request. Whatever the interrupt is serviced or not, the instruction following the IDLE instruction will be executed after return from interrupt (RETI) instruction, then the CPU resumes the normal program.

20.2 Power down mode

Power Down mode starts by running PWRDN protected instruction. Internal clock is stopped, all MCU parts are on hold including the watchdog timer. The only exception could be the Real Time Clock if opportunely programmed and one of the two oscillator circuits as a consequence (either the main or the 32 kHz on-chip oscillator).

When Real Time Clock module is used, when the device is in Power Down mode a reference clock is needed. In this case, two possible configurations may be selected by the user application according to the desired level of power reduction:

- A 32 kHz crystal is connected to the on-chip low-power oscillator (pins XTAL3 / XTAL4) and running. In this case the main oscillator is stopped when Power Down mode is entered, while the Real Time Clock continue counting using 32 kHz clock signal as reference. The presence of a running low-power oscillator is detected after the Power-on: this clock is immediately assumed (if present, or as soon as it is detected) as reference for the Real Time Clock counter and it will be maintained forever (unless specifically disabled via software).
- Only the main oscillator is running (XTAL1 / XTAL2 pins). In this case the main oscillator is not stopped when Power Down is entered, and the Real Time Clock continue counting using the main oscillator clock signal as reference.

There are two different operating Power Down modes: protected mode and interruptible mode.

Before entering Power Down mode (by executing the instruction PWRDN), bit VREGOFF in XMISC register must be set.

Note: Leaving the main voltage regulator active during Power Down may lead to unexpected behavior (ex: CPU wake-up) and power consumption higher than what specified.

20.2.1 Protected power down mode

This mode is selected when PWDCFG (bit 5) of SYSCON register is cleared. The Protected Power Down mode is only activated if the $\overline{\text{NMI}}$ pin is pulled low when executing PWRDN instruction (this means that the PWRDN instruction belongs to the $\overline{\text{NMI}}$ software routine). This mode is only deactivated with an external hardware reset on $\overline{\text{RSTIN}}$ pin.

20.2.2 Interruptible power down mode

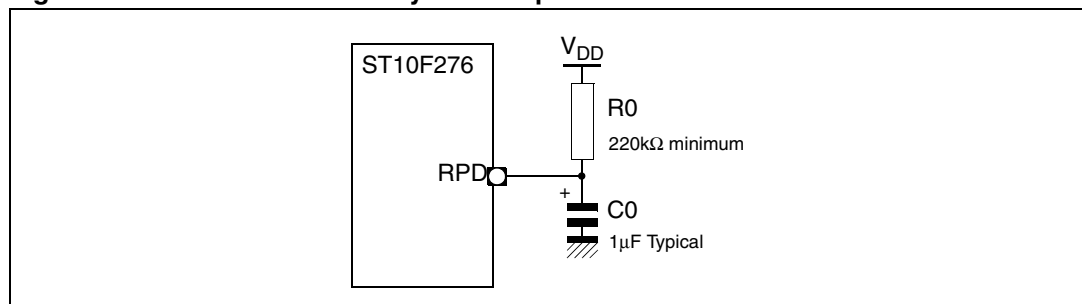
This mode is selected when PWDCFG (bit 5) of SYSCON register is set.

The Interruptible Power Down mode is only activated if all the enabled Fast External Interrupt pins are in their inactive level.

This mode is deactivated with an external reset applied to $\overline{\text{RSTIN}}$ pin or with an interrupt request applied to one of the Fast External Interrupt pins, or with an interrupt generated by the Real Time Clock, or with an interrupt generated by the activity on CAN's and I²C module interfaces. To allow the internal PLL and clock to stabilize, the $\overline{\text{RSTIN}}$ pin must be held low according the recommendations described in [Chapter 19: System reset](#).

An external RC circuit must be connected to RPD pin, as shown in the [Figure 43](#).

Figure 43. External RC circuitry on RPD pin



To exit Power Down mode with an external interrupt, an EXxIN (x = 7...0) pin has to be asserted for at least 40ns.

20.3 Stand-by mode

In Stand-by mode, it is possible to turn off the main V_{DD} provided that V_{STBY} is available through the dedicated pin of the ST10F276.

To enter Stand-by mode it is mandatory to held the device under reset: once the device is under reset, the RAM is disabled (see XRAM2EN bit of XPERCON register), and its digital interface is frozen in order to avoid any kind of data corruption.

A dedicated embedded low-power voltage regulator is implemented to generate the internal low voltage supply (about 1.65V in Stand-by mode) to bias all those circuits that shall remain active: the portion of XRAM (16Kbytes for ST10F273E), the RTC counters and 32 kHz on-chip oscillator amplifier.

In normal running mode (that is when main V_{DD} is on) the V_{STBY} pin can be tied to V_{SS} during reset to exercise the \overline{EA} functionality associated with the same pin: the voltage supply for the circuitries which are usually biased with V_{STBY} (see in particular the 32 kHz oscillator used in conjunction with Real Time Clock module), is granted by the active main V_{DD} .

It must be noted that Stand-by Mode can generate problems associated with the usage of different power supplies in CMOS systems; particular attention must be paid when the ST10F276 I/O lines are interfaced with other external CMOS integrated circuits: if V_{DD} of ST10F276 becomes (for example in Stand-by Mode) lower than the output level forced by the I/O lines of these external integrated circuits, the ST10F276 could be directly powered through the inherent diode existing on ST10F276 output driver circuitry. The same is valid for ST10F276 interfaced to active/inactive communication buses during Stand-by mode: current injection can be generated through the inherent diode.

Furthermore, the sequence of turning on/off of the different voltage could be critical for the system (not only for the ST10F276 device). The device Stand-by mode current (I_{STBY}) may vary while V_{DD} to V_{STBY} (and vice versa) transition occurs: some current flows between V_{DD} and V_{STBY} pins. System noise on both V_{DD} and V_{STBY} can contribute to increase this phenomenon.

20.3.1 Entering stand-by mode

As already said, to enter Stand-by Mode XRAM2EN bit in the XPERCON Register must be cleared: this allows to freeze immediately the RAM interface, avoiding any data corruption. As a consequence of a RESET event, the RAM Power Supply is switched to the internal low-voltage supply V_{18SB} (derived from V_{STBY} through the low-power voltage regulator). The RAM interface will remain frozen until the bit XRAM2EN is set again by software initialization routine (at next exit from main V_{DD} power-on reset sequence).

Since V_{18} is falling down (as a consequence of V_{DD} turning off), it can happen that the XRAM2EN bit is no longer able to guarantee its content (logic "0"), being the XPERCON Register powered by internal V_{18} . This does not generate any problem, because the Stand-by Mode switching dedicated circuit continues to confirm the RAM interface freezing, irrespective the XRAM2EN bit content; XRAM2EN bit status is considered again when internal V_{18} comes back over internal stand-by reference V_{18SB} .

If internal V_{18} becomes lower than internal stand-by reference (V_{18SB}) of about 0.3 to 0.45V with bit XRAM2EN set, the RAM Supply switching circuit is not active: in case of a temporary drop on internal V_{18} voltage versus internal V_{18SB} during normal code execution, no spurious Stand-by Mode switching can occur (the RAM is not frozen and can still be accessed).

The ST10F276 Core module, generating the RAM control signals, is powered by internal V_{18} supply; during turning off transient these control signals follow the V_{18} , while RAM is switched to V_{18SB} internal reference. It could happen that a high level of RAM write strobe from ST10F276 Core (active low signal) is low enough to be recognized as a logic "0" by the RAM interface (due to V_{18} lower than V_{18SB}): The bus status could contain a valid address for the RAM and an unwanted data corruption could occur. For this reason, an extra interface, powered by the switched supply, is used to prevent the RAM from this kind of potential corruption mechanism.

Warning: During power-off phase, it is important that the external hardware maintains a stable ground level on $\overline{\text{RSTIN}}$ pin, without any glitch, in order to avoid spurious exiting from reset status with unstable power supply.

20.3.2 Exiting stand-by mode

After the system has entered the Stand-by Mode, the procedure to exit this mode consists of a standard Power-on sequence, with the only difference that the RAM is already powered through V_{18SB} internal reference (derived from V_{STBY} pin external voltage).

It is recommended to held the device under RESET ($\overline{\text{RSTIN}}$ pin forced low) until external V_{DD} voltage pin is stable. Even though, at the very beginning of the power-on phase, the device is maintained under reset by the internal low voltage detector circuit (implemented inside the main voltage regulator) till the internal V_{18} becomes higher than about 1.0V, **there is no warranty that the device stays under reset status if $\overline{\text{RSTIN}}$ is at high level during power ramp up. So, it is important the external hardware is able to guarantee a stable ground level on $\overline{\text{RSTIN}}$ along the power-on phase, without any temporary glitch.**

The external hardware shall be responsible to drive low the $\overline{\text{RSTIN}}$ pin until the V_{DD} is stable, even though the internal LVD is active.

Once the internal Reset signal goes low, the RAM (still frozen) power supply is switched to the main V_{18} .

At this time, everything becomes stable, and the execution of the initialization routines can start: XRAM2EN bit can be set, enabling the RAM.

20.3.3 Real time clock and stand-by mode

When Stand-by mode is entered (turning off the main supply V_{DD}), the Real Time Clock counting can be maintained running in case the on-chip 32 kHz oscillator is used to provide the reference to the counter. This is not possible if the main oscillator is used as reference for the counter: Being the main oscillator powered by V_{DD} , once this is switched off, the oscillator is stopped.

20.3.4 Power reduction modes summary

In the following [Table 63: Power reduction modes summary](#), a summary of the different Power reduction modes is reported.

Table 63. Power reduction modes summary

| Mode | V _{DD} | V _{STBY} | CPU | Peripherals | RTC | Main OSC | 32 kHz OSC | STBY XRAM | XRAM |
|------------|-----------------|-------------------|-----|-------------|-----|----------|------------|-----------|--------|
| Idle | on | on | off | on | off | run | off | biased | biased |
| | on | on | off | on | on | run | on | biased | biased |
| Power Down | on | on | off | off | off | off | off | biased | biased |
| | on | on | off | off | on | on | off | biased | biased |
| | on | on | off | off | on | off | on | biased | biased |
| Stand-by | off | on | off | off | off | off | off | biased | off |
| | off | on | off | off | on | off | on | biased | off |

21 Programmable output clock divider

A specific register mapped on the XBUS allows to choose the division factor on the CLKOUT signal (P3.15). This register is mapped on X-Miscellaneous memory address range.

When CLKOUT function is enabled by setting bit CLKEN of register SYSCON, by default the CPU clock is output on P3.15. Setting bit XMISCEN of register XPERCON and bit XPEN of register SYSCON, it is possible to program the clock prescaling factor: in this way on P3.15 a prescaled value of the CPU clock can be output.

When CLKOUT function is not enabled (bit CLKEN of register SYSCON cleared), P3.15 does not output any clock signal, even though XCLKOUTDIV register is programmed.

22 Register set

This section summarizes all registers implemented in the ST10F276 and explains the description format used in the chapters to describe the function and layout of the SFRs. For easy reference, the registers (except for GPRs) are sorted in two ways:

- Sorted by address, to check which register is referenced by a given address.
- Sorted by register name, to find the location of a specific register.

22.1 Register description format

Throughout the document, the function and the layout of the different registers is described in a specific format. The example below explains this format.

A word register is displayed as:

| REG_NAME (A16h / A8h) | | | | | | SFR/ESFR/XBUS | | | | | | Reset value: ****h: | | | |
|-----------------------|------|------|------|------|------------|---------------|-----------|---------|--------|----------|----|---------------------|----------|----|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| res. | res. | res. | res. | res. | write only | hw bit | read only | std bit | hw bit | bitfield | | | bitfield | | |
| | | | | | W | RW | R | RW | RW | | RW | | | RW | |

Table 64. Description

| Bit | Function |
|-----------------|---|
| Bit(field) name | Explanation of bit(field) name Description of the functions controlled by this bit(field). |

A byte register is displayed as:

| REG_NAME (A16h / A8h) | | | | | | SFR/ESFR/XBUS | | | | | | Reset value: -- **h: | | | |
|-----------------------|----|----|----|----|----|---------------|---|---------|--------|-----------|---|----------------------|-----------|----|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | - | - | - | - | - | std bit | hw bit | bit field | | | bit field | | |
| | | | | | | | | RW | RW | RW | | | | RW | |

Elements:

| | |
|---------------|--|
| REG_NAME | This register's name |
| A16h / A8h | Long 16-bit address / Short 8-bit address |
| SFR/ESFR/XBUS | Register space (SFR, ESFR or XBUS Register) |
| (*) ** | Register contents after reset |
| | 0/1 : defined |
| | X : undefined (undefined ('X') after power up) |
| | U : unchanged |
| hwbit | Bit that is set/cleared by hardware is written in bold |

22.2 General purpose registers (GPRs)

The GPRs form the register bank that the CPU works with. This register bank may be located anywhere within the internal RAM via the Context Pointer (CP). Due to the addressing mechanism, GPR banks reside only within the internal RAM. All GPRs are bit-addressable.

Table 65. General purpose registers (GPRs)

| Name | Physical address | 8-bit address | Description | Reset value |
|------|------------------|---------------|---|-------------|
| R0 | (CP) + 0 | F0h | CPU general purpose (word) register R0 | UUUUh |
| R1 | (CP) + 2 | F1h | CPU general purpose (word) register R1 | UUUUh |
| R2 | (CP) + 4 | F2h | CPU general purpose (word) register R2 | UUUUh |
| R3 | (CP) + 6 | F3h | CPU general purpose (word) register R3 | UUUUh |
| R4 | (CP) + 8 | F4h | CPU general purpose (word) register R4 | UUUUh |
| R5 | (CP) + 10 | F5h | CPU general purpose (word) register R5 | UUUUh |
| R6 | (CP) + 12 | F6h | CPU general purpose (word) register R6 | UUUUh |
| R7 | (CP) + 14 | F7h | CPU general purpose (word) register R7 | UUUUh |
| R8 | (CP) + 16 | F8h | CPU general purpose (word) register R8 | UUUUh |
| R9 | (CP) + 18 | F9h | CPU general purpose (word) register R9 | UUUUh |
| R10 | (CP) + 20 | FAh | CPU general purpose (word) register R10 | UUUUh |
| R11 | (CP) + 22 | FBh | CPU general purpose (word) register R11 | UUUUh |
| R12 | (CP) + 24 | FCh | CPU general purpose (word) register R12 | UUUUh |
| R13 | (CP) + 26 | FDh | CPU general purpose (word) register R13 | UUUUh |
| R14 | (CP) + 28 | FEh | CPU general purpose (word) register R14 | UUUUh |
| R15 | (CP) + 30 | FFh | CPU general purpose (word) register R15 | UUUUh |

The first 8 GPRs (R7...R0) may also be accessed byte-wise. Other than with SFRs, writing to a GPR byte does not affect the other byte of the respective GPR. The respective halves of the byte-accessible registers have special names:

Table 66. General purpose registers (GPRs) byte-wise addressing

| Name | Physical address | 8-bit address | Description | Reset value |
|------|------------------|---------------|---|-------------|
| RL0 | (CP) + 0 | F0h | CPU general purpose (byte) register RL0 | UUh |
| RH0 | (CP) + 1 | F1h | CPU general purpose (byte) register RH0 | UUh |
| RL1 | (CP) + 2 | F2h | CPU general purpose (byte) register RL1 | UUh |
| RH1 | (CP) + 3 | F3h | CPU general purpose (byte) register RH1 | UUh |
| RL2 | (CP) + 4 | F4h | CPU general purpose (byte) register RL2 | UUh |
| RH2 | (CP) + 5 | F5h | CPU general purpose (byte) register RH2 | UUh |

Table 66. General purpose registers (GPRs) bitwise addressing

| Name | Physical address | 8-bit address | Description | Reset value |
|------|------------------|---------------|---|-------------|
| RL0 | (CP) + 0 | F0h | CPU general purpose (byte) register RL0 | UUh |
| RL3 | (CP) + 6 | F6h | CPU general purpose (byte) register RL3 | UUh |
| RH3 | (CP) + 7 | F7h | CPU general purpose (byte) register RH3 | UUh |
| RL4 | (CP) + 8 | F8h | CPU general purpose (byte) register RL4 | UUh |
| RH4 | (CP) + 9 | F9h | CPU general purpose (byte) register RH4 | UUh |
| RL5 | (CP) + 10 | FAh | CPU general purpose (byte) register RL5 | UUh |
| RH5 | (CP) + 11 | FBh | CPU general purpose (byte) register RH5 | UUh |
| RL6 | (CP) + 12 | FCh | CPU general purpose (byte) register RL6 | UUh |
| RH6 | (CP) + 13 | FDh | CPU general purpose (byte) register RH6 | UUh |
| RL7 | (CP) + 14 | FEh | CPU general purpose (byte) register RL7 | UUh |
| RH7 | (CP) + 15 | FFh | CPU general purpose (byte) register RH7 | UUh |

22.3 Special function registers ordered by name

The following table lists in alphabetical order all SFRs which are implemented in the ST10F276.

Bit-addressable SFRs are marked with the letter “b” in column “Name”.

SFRs within the **Extended SFR-Space** (ESFRs) are marked with the letter “E” in column “Physical Address”.

Table 67. Special function registers ordered by address

| Name | Physical address | 8-bit address | Description | Reset value |
|------------------|------------------|---------------|--|-------------|
| ADCIC b | FF98h | CCh | A/D converter end of conversion interrupt control register | - - 00h |
| ADCON b | FFA0h | D0h | A/D converter control register | 0000h |
| ADDAT | FEA0h | 50h | A/D converter result register | 0000h |
| ADDAT2 | F0A0h E | 50h | A/D converter 2 result register | 0000h |
| ADDRSEL1 | FE18h | 0Ch | Address select register 1 | 0000h |
| ADDRSEL2 | FE1Ah | 0Dh | Address select register 2 | 0000h |
| ADDRSEL3 | FE1Ch | 0Eh | Address select register 3 | 0000h |
| ADDRSEL4 | FE1Eh | 0Fh | Address select register 4 | 0000h |
| ADEIC b | FF9Ah | CDh | A/D converter overrun error interrupt control register | - - 00h |
| BUSCON0 b | FF0Ch | 86h | Bus configuration register 0 | 0xx0h |
| BUSCON1 b | FF14h | 8Ah | Bus configuration register 1 | 0000h |
| BUSCON2 b | FF16h | 8Bh | Bus configuration register 2 | 0000h |
| BUSCON3 b | FF18h | 8Ch | Bus configuration register 3 | 0000h |
| BUSCON4 b | FF1Ah | 8Dh | Bus configuration register 4 | 0000h |
| CAPREL | FE4Ah | 25h | GPT2 capture/reload register | 0000h |
| CC0 | FE80h | 40h | CAPCOM register 0 | 0000h |
| CC0IC b | FF78h | BCh | CAPCOM register 0 interrupt control register | - - 00h |
| CC1 | FE82h | 41h | CAPCOM register 1 | 0000h |
| CC10 | FE94h | 4Ah | CAPCOM register 10 | 0000h |
| CC10IC b | FF8Ch | C6h | CAPCOM register 10 interrupt control register | - - 00h |
| CC11 | FE96h | 4Bh | CAPCOM register 11 | 0000h |
| CC11IC b | FF8Eh | C7h | CAPCOM register 11 interrupt control register | - - 00h |
| CC12 | FE98h | 4Ch | CAPCOM register 12 | 0000h |
| CC12IC b | FF90h | C8h | CAPCOM register 12 interrupt control register | - - 00h |
| CC13 | FE9Ah | 4Dh | CAPCOM register 13 | 0000h |
| CC13IC b | FF92h | C9h | CAPCOM register 13 interrupt control register | - - 00h |
| CC14 | FE9Ch | 4Eh | CAPCOM register 14 | 0000h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|---|-------------|
| CC14IC b | FF94h | CAh | CAPCOM register 14 interrupt control register | - - 00h |
| CC15 | FE9Eh | 4Fh | CAPCOM register 15 | 0000h |
| CC15IC b | FF96h | CBh | CAPCOM register 15 interrupt control register | - - 00h |
| CC16 | FE60h | 30h | CAPCOM register 16 | 0000h |
| CC16IC b | F160h E | B0h | CAPCOM register 16 interrupt control register | - - 00h |
| CC17 | FE62h | 31h | CAPCOM register 17 | 0000h |
| CC17IC b | F162h E | B1h | CAPCOM register 17 interrupt control register | - - 00h |
| CC18 | FE64h | 32h | CAPCOM register 18 | 0000h |
| CC18IC b | F164h E | B2h | CAPCOM register 18 interrupt control register | - - 00h |
| CC19 | FE66h | 33h | CAPCOM register 19 | 0000h |
| CC19IC b | F166h E | B3h | CAPCOM register 19 interrupt control register | - - 00h |
| CC1IC b | FF7Ah | BDh | CAPCOM register 1 interrupt control register | - - 00h |
| CC2 | FE84h | 42h | CAPCOM register 2 | 0000h |
| CC20 | FE68h | 34h | CAPCOM register 20 | 0000h |
| CC20IC b | F168h E | B4h | CAPCOM register 20 interrupt control register | - - 00h |
| CC21 | FE6Ah | 35h | CAPCOM register 21 | 0000h |
| CC21IC b | F16Ah E | B5h | CAPCOM register 21 interrupt control register | - - 00h |
| CC22 | FE6Ch | 36h | CAPCOM register 22 | 0000h |
| CC22IC b | F16Ch E | B6h | CAPCOM register 22 interrupt control register | - - 00h |
| CC23 | FE6Eh | 37h | CAPCOM register 23 | 0000h |
| CC23IC b | F16Eh E | B7h | CAPCOM register 23 interrupt control register | - - 00h |
| CC24 | FE70h | 38h | CAPCOM register 24 | 0000h |
| CC24IC b | F170h E | B8h | CAPCOM register 24 interrupt control register | - - 00h |
| CC25 | FE72h | 39h | CAPCOM register 25 | 0000h |
| CC25IC b | F172h E | B9h | CAPCOM register 25 interrupt control register | - - 00h |
| CC26 | FE74h | 3Ah | CAPCOM register 26 | 0000h |
| CC26IC b | F174h E | BAh | CAPCOM register 26 interrupt control register | - - 00h |
| CC27 | FE76h | 3Bh | CAPCOM register 27 | 0000h |
| CC27IC b | F176h E | BBh | CAPCOM register 27 interrupt control register | - - 00h |
| CC28 | FE78h | 3Ch | CAPCOM register 28 | 0000h |
| CC28IC b | F178h E | BCh | CAPCOM register 28 interrupt control register | - - 00h |
| CC29 | FE7Ah | 3Dh | CAPCOM register 29 | 0000h |
| CC29IC b | F184h E | C2h | CAPCOM register 29 interrupt control register | - - 00h |
| CC2IC b | FF7Ch | BEh | CAPCOM register 2 interrupt control register | - - 00h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|---|-------------|
| CC3 | FE86h | 43h | CAPCOM register 3 | 0000h |
| CC30 | FE7Ch | 3Eh | CAPCOM register 30 | 0000h |
| CC30IC b | F18Ch E | C6h | CAPCOM register 30 interrupt control register | - - 00h |
| CC31 | FE7Eh | 3Fh | CAPCOM register 31 | 0000h |
| CC31IC b | F194h E | CAh | CAPCOM register 31 interrupt control register | - - 00h |
| CC3IC b | FF7Eh | BFh | CAPCOM register 3 interrupt control register | - - 00h |
| CC4 | FE88h | 44h | CAPCOM register 4 | 0000h |
| CC4IC b | FF80h | C0h | CAPCOM register 4 interrupt control register | - - 00h |
| CC5 | FE8Ah | 45h | CAPCOM register 5 | 0000h |
| CC5IC b | FF82h | C1h | CAPCOM register 5 interrupt control register | - - 00h |
| CC6 | FE8Ch | 46h | CAPCOM register 6 | 0000h |
| CC6IC b | FF84h | C2h | CAPCOM register 6 interrupt control register | - - 00h |
| CC7 | FE8Eh | 47h | CAPCOM register 7 | 0000h |
| CC7IC b | FF86h | C3h | CAPCOM register 7 interrupt control register | - - 00h |
| CC8 | FE90h | 48h | CAPCOM register 8 | 0000h |
| CC8IC b | FF88h | C4h | CAPCOM register 8 interrupt control register | - - 00h |
| CC9 | FE92h | 49h | CAPCOM register 9 | 0000h |
| CC9IC b | FF8Ah | C5h | CAPCOM register 9 interrupt control register | - - 00h |
| CCM0 b | FF52h | A9h | CAPCOM mode control register 0 | 0000h |
| CCM1 b | FF54h | AAh | CAPCOM mode control register 1 | 0000h |
| CCM2 b | FF56h | ABh | CAPCOM mode control register 2 | 0000h |
| CCM3 b | FF58h | ACH | CAPCOM mode control register 3 | 0000h |
| CCM4 b | FF22h | 91h | CAPCOM mode control register 4 | 0000h |
| CCM5 b | FF24h | 92h | CAPCOM mode control register 5 | 0000h |
| CCM6 b | FF26h | 93h | CAPCOM mode control register 6 | 0000h |
| CCM7 b | FF28h | 94h | CAPCOM mode control register 7 | 0000h |
| CP | FE10h | 08h | CPU context pointer register | FC00h |
| CRIC b | FF6Ah | B5h | GPT2 CAPREL interrupt control register | - - 00h |
| CSP | FE08h | 04h | CPU code segment pointer register (read-only) | 0000h |
| DP0H b | F102h E | 81h | P0H direction control register | - - 00h |
| DP0L b | F100h E | 80h | P0L direction control register | - - 00h |
| DP1H b | F106h E | 83h | P1H direction control register | - - 00h |
| DP1L b | F104h E | 82h | P1L direction control register | - - 00h |
| DP2 b | FFC2h | E1h | Port 2 direction control register | 0000h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|---|-------------|
| DP3 b | FFC6h | E3h | Port 3 direction control register | 0000h |
| DP4 b | FFCAh | E5h | Port 4 direction control register | - - 00h |
| DP6 b | FFCEh | E7h | Port 6 direction control register | - - 00h |
| DP7 b | FFD2h | E9h | Port 7 direction control register | - - 00h |
| DP8 b | FFD6h | EBh | Port 8 direction control register | - - 00h |
| DPP0 | FE00h | 00h | CPU data page pointer 0 register (10-bit) | 0000h |
| DPP1 | FE02h | 01h | CPU data page pointer 1 register (10-bit) | 0001h |
| DPP2 | FE04h | 02h | CPU data page pointer 2 register (10-bit) | 0002h |
| DPP3 | FE06h | 03h | CPU data page pointer 3 register (10-bit) | 0003h |
| EMUCON | FE0Ah | 05h | Emulation control register | - - XXh |
| EXICON b | F1C0h E | E0h | External interrupt control register | 0000h |
| EXISEL b | F1DAh E | EDh | External interrupt source selection register | 0000h |
| IDCHIP | F07Ch E | 3Eh | Device identifier register (n is the device revision) | 114nh |
| IDMANUF | F07Eh E | 3Fh | Manufacturer identifier register | 0403h |
| IDMEM | F07Ah E | 3Dh | On-chip memory identifier register | 30D0h |
| IDPROG | F078h E | 3Ch | Programming voltage identifier register | 0040h |
| IDX0 b | FF08h | 84h | MAC unit address pointer 0 | 0000h |
| IDX1 b | FF0Ah | 85h | MAC unit address pointer 1 | 0000h |
| MAH | FE5Eh | 2Fh | MAC unit accumulator - High word | 0000h |
| MAL | FE5Ch | 2Eh | MAC unit accumulator - Low word | 0000h |
| MCW b | FFDCh | EEh | MAC unit control word | 0000h |
| MDC b | FF0Eh | 87h | CPU multiply divide control register | 0000h |
| MDH | FE0Ch | 06h | CPU multiply divide register – High word | 0000h |
| MDL | FE0Eh | 07h | CPU multiply divide register – Low word | 0000h |
| MRW b | FFDAh | EDh | MAC unit repeat word | 0000h |
| MSW b | FFDEh | EFh | MAC unit status word | 0200h |
| ODP2 b | F1C2h E | E1h | Port2 open drain control register | 0000h |
| ODP3 b | F1C6h E | E3h | Port3 open drain control register | 0000h |
| ODP4 b | F1CAh E | E5h | Port4 open drain control register | - - 00h |
| ODP6 b | F1CEh E | E7h | Port6 open drain control register | - - 00h |
| ODP7 b | F1D2h E | E9h | Port7 open drain control register | - - 00h |
| ODP8 b | F1D6h E | EBh | Port8 open drain control register | - - 00h |
| ONES b | FF1Eh | 8Fh | Constant value 1's register (read-only) | FFFFh |
| P0H b | FF02h | 81h | Port0 high register (upper half of PORT0) | - - 00h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|------------------|------------------|---------------|---|-------------|
| P0L b | FF00h | 80h | Port0 low register (lower half of PORT0) | - - 00h |
| P1H b | FF06h | 83h | Port1 high register (upper half of PORT1) | - - 00h |
| P1L b | FF04h | 82h | Port1 low register (lower half of PORT1) | - - 00h |
| P2 b | FFC0h | E0h | Port 2 register | 0000h |
| P3 b | FFC4h | E2h | Port 3 register | 0000h |
| P4 b | FFC8h | E4h | Port 4 register (8-bit) | - - 00h |
| P5 b | FFA2h | D1h | Port 5 register (read-only) | XXXXh |
| P5DIDIS b | FFA4h | D2h | Port 5 digital disable register | 0000h |
| P6 b | FFCCh | E6h | Port 6 register (8-bit) | - - 00h |
| P7 b | FFD0h | E8h | Port 7 register (8-bit) | - - 00h |
| P8 b | FFD4h | EAh | Port 8 register (8-bit) | - - 00h |
| PECC0 | FEC0h | 60h | PEC channel 0 control register | 0000h |
| PECC1 | FEC2h | 61h | PEC channel 1 control register | 0000h |
| PECC2 | FEC4h | 62h | PEC channel 2 control register | 0000h |
| PECC3 | FEC6h | 63h | PEC channel 3 control register | 0000h |
| PECC4 | FEC8h | 64h | PEC channel 4 control register | 0000h |
| PECC5 | FECAh | 65h | PEC channel 5 control register | 0000h |
| PECC6 | FECCh | 66h | PEC channel 6 control register | 0000h |
| PECC7 | FECEh | 67h | PEC channel 7 control register | 0000h |
| PICON b | F1C4h E | E2h | Port input threshold control register | - - 00h |
| PP0 | F038h E | 1Ch | PWM module period register 0 | 0000h |
| PP1 | F03Ah E | 1Dh | PWM module period register 1 | 0000h |
| PP2 | F03Ch E | 1Eh | PWM module period register 2 | 0000h |
| PP3 | F03Eh E | 1Fh | PWM module period register 3 | 0000h |
| PSW b | FF10h | 88h | CPU program status word | 0000h |
| PT0 | F030h E | 18h | PWM module up/down counter 0 | 0000h |
| PT1 | F032h E | 19h | PWM module up/down counter 1 | 0000h |
| PT2 | F034h E | 1Ah | PWM module up/down counter 2 | 0000h |
| PT3 | F036h E | 1Bh | PWM module up/down counter 3 | 0000h |
| PW0 | FE30h | 18h | PWM module pulse width register 0 | 0000h |
| PW1 | FE32h | 19h | PWM module pulse width register 1 | 0000h |
| PW2 | FE34h | 1Ah | PWM module pulse width register 2 | 0000h |
| PW3 | FE36h | 1Bh | PWM module pulse width register 3 | 0000h |
| PWMCON0 b | FF30h | 98h | PWM module control register 0 | 0000h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|------------------|------------------|---------------|---|-------------|
| PWMCON1 b | FF32h | 99h | PWM module control register 1 | 0000h |
| PWMIC b | F17Eh E | BFh | PWM Module interrupt control register | - - 00h |
| QR0 | F004h E | 02h | MAC unit offset register R0 | 0000h |
| QR1 | F006h E | 03h | MAC unit offset register R1 | 0000h |
| QX0 | F000h E | 00h | MAC unit Offset Register X0 | 0000h |
| QX1 | F002h E | 01h | MAC unit offset register X1 | 0000h |
| RP0H b | F108h E | 84h | System start-up configuration register (read-only) | - - XXh |
| S0BG | FEB4h | 5Ah | Serial channel 0 baud rate generator reload register | 0000h |
| S0CON b | FFB0h | D8h | Serial channel 0 control register | 0000h |
| S0EIC b | FF70h | B8h | Serial channel 0 error interrupt control register | - - 00h |
| S0RBUF | FEB2h | 59h | Serial channel 0 receive buffer register (read-only) | - - XXh |
| S0RIC b | FF6Eh | B7h | Serial channel 0 receive interrupt control register | - - 00h |
| S0TBIC b | F19Ch E | CEh | Serial channel 0 transmit buffer interrupt control register | - - 00h |
| S0TBUF | FEB0h | 58h | Serial channel 0 transmit buffer register (write-only) | 0000h |
| S0TIC b | FF6Ch | B6h | Serial channel 0 transmit interrupt control register | - - 00h |
| SP | FE12h | 09h | CPU system stack pointer register | FC00h |
| SSCBR | F0B4h E | 5Ah | SSC baud rate register | 0000h |
| SSCCON b | FFB2h | D9h | SSC control register | 0000h |
| SSCEIC b | FF76h | BBh | SSC error interrupt control register | - - 00h |
| SSCRB | F0B2h E | 59h | SSC receive buffer (read-only) | XXXXh |
| SSCRIC b | FF74h | BAh | SSC receive interrupt control register | - - 00h |
| SSCTB | F0B0h E | 58h | SSC transmit buffer (write-only) | 0000h |
| SSCTIC b | FF72h | B9h | SSC transmit interrupt control register | - - 00h |
| STKOV | FE14h | 0Ah | CPU stack overflow pointer register | FA00h |
| STKUN | FE16h | 0Bh | CPU stack underflow pointer register | FC00h |
| SYSCON b | FF12h | 89h | CPU system configuration register | 0xx0h |
| T0 | FE50h | 28h | CAPCOM timer 0 register | 0000h |
| T01CON b | FF50h | A8h | CAPCOM timer 0 and timer 1 control register | 0000h |
| T0IC b | FF9Ch | CEh | CAPCOM timer 0 interrupt control register | - - 00h |
| T0REL | FE54h | 2Ah | CAPCOM timer 0 reload register | 0000h |
| T1 | FE52h | 29h | CAPCOM timer 1 register | 0000h |
| T1IC b | FF9Eh | CFh | CAPCOM timer 1 interrupt control register | - - 00h |
| T1REL | FE56h | 2Bh | CAPCOM timer 1 reload register | 0000h |

Table 67. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|---|-------------|
| T2 | FE40h | 20h | GPT1 timer 2 register | 0000h |
| T2CON b | FF40h | A0h | GPT1 timer 2 control register | 0000h |
| T2IC b | FF60h | B0h | GPT1 timer 2 interrupt control register | - - 00h |
| T3 | FE42h | 21h | GPT1 timer 3 register | 0000h |
| T3CON b | FF42h | A1h | GPT1 timer 3 control register | 0000h |
| T3IC b | FF62h | B1h | GPT1 timer 3 interrupt control register | - - 00h |
| T4 | FE44h | 22h | GPT1 timer 4 register | 0000h |
| T4CON b | FF44h | A2h | GPT1 timer 4 control register | 0000h |
| T4IC b | FF64h | B2h | GPT1 timer 4 interrupt control register | - - 00h |
| T5 | FE46h | 23h | GPT2 timer 5 register | 0000h |
| T5CON b | FF46h | A3h | GPT2 timer 5 control register | 0000h |
| T5IC b | FF66h | B3h | GPT2 timer 5 interrupt control register | - - 00h |
| T6 | FE48h | 24h | GPT2 timer 6 register | 0000h |
| T6CON b | FF48h | A4h | GPT2 timer 6 control register | 0000h |
| T6IC b | FF68h | B4h | GPT2 timer 6 interrupt control register | - - 00h |
| T7 | F050h E | 28h | CAPCOM timer 7 register | 0000h |
| T78CON b | FF20h | 90h | CAPCOM timer 7 and 8 control register | 0000h |
| T7IC b | F17Ah E | BDh | CAPCOM timer 7 interrupt control register | - - 00h |
| T7REL | F054h E | 2Ah | CAPCOM timer 7 reload register | 0000h |
| T8 | F052h E | 29h | CAPCOM timer 8 register | 0000h |
| T8IC b | F17Ch E | BEh | CAPCOM timer 8 interrupt control register | - - 00h |
| T8REL | F056h E | 2Bh | CAPCOM timer 8 reload register | 0000h |
| TFR b | FFACh | D6h | Trap flag register | 0000h |
| WDT | FEAEh | 57h | Watchdog timer register (read-only) | 0000h |
| WDTCON b | FFAEh | D7h | Watchdog timer control register | 00xxh |
| XADRS3 | F01Ch E | 0Eh | XPER address select register 3 | 800Bh |
| XP0IC b | F186h E | C3h | See Section 8.1 | - - 00h |
| XP1IC b | F18Eh E | C7h | See Section 8.1 | - - 00h |
| XP2IC b | F196h E | CBh | See Section 8.1 | - - 00h |
| XP3IC b | F19Eh E | CFh | See Section 8.1 | - - 00h |
| XPERCON | F024h E | 12h | XPER configuration register | - - 05h |
| ZEROS b | FF1Ch | 8Eh | Constant value 0's register (read-only) | 0000h |

- Note:
- 1 The system configuration is selected during reset. SYSCON reset value is 0000 0xx0 x000 0000b.
 - 2 Reset Value depends on different triggered reset event.
 - 3 The XPnIC Interrupt Control Registers control interrupt requests from integrated X-Bus peripherals. Some software controlled interrupt requests may be generated by setting the XPnIR bits (of XPnIC register) of the unused X-Peripheral nodes.

22.4 Special function registers ordered by address

The following table lists by order of their physical addresses all SFRs which are implemented in the ST10F276 .

Bit-addressable SFRs are marked with the letter “b” in column “Name”.

SFRs within the **Extended SFR-Space** (ESFRs) are marked with the letter “E” in column “Physical Address”.

Table 68. Special function registers ordered by address

| Name | Physical address | 8-bit address | Description | Reset value |
|---------|------------------|---------------|---|-------------|
| QX0 | F000h E | 00h | MAC unit offset register X0 | 0000h |
| QX1 | F002h E | 01h | MAC unit offset register X1 | 0000h |
| QR0 | F004h E | 02h | MAC unit offset register R0 | 0000h |
| QR1 | F006h E | 03h | MAC unit offset register R1 | 0000h |
| XADRS3 | F01Ch E | 0Eh | XPER address select register 3 | 800Bh |
| XPERCON | F024h E | 12h | XPER configuration register | - - 05h |
| PT0 | F030h E | 18h | PWM module up/down counter 0 | 0000h |
| PT1 | F032h E | 19h | PWM module up/down counter 1 | 0000h |
| PT2 | F034h E | 1Ah | PWM module up/down counter 2 | 0000h |
| PT3 | F036h E | 1Bh | PWM module up/down counter 3 | 0000h |
| PP0 | F038h E | 1Ch | PWM module period register 0 | 0000h |
| PP1 | F03Ah E | 1Dh | PWM module period register 1 | 0000h |
| PP2 | F03Ch E | 1Eh | PWM module period register 2 | 0000h |
| PP3 | F03Eh E | 1Fh | PWM module period register 3 | 0000h |
| T7 | F050h E | 28h | CAPCOM timer 7 register | 0000h |
| T8 | F052h E | 29h | CAPCOM timer 8 register | 0000h |
| T7REL | F054h E | 2Ah | CAPCOM timer 7 reload register | 0000h |
| T8REL | F056h E | 2Bh | CAPCOM timer 8 reload register | 0000h |
| IDPROG | F078h E | 3Ch | Programming voltage identifier register | 0040h |
| IDMEM | F07Ah E | 3Dh | On-chip memory identifier register | 30D0h |
| IDCHIP | F07Ch E | 3Eh | Device identifier register (n is the device revision) | 114nh |
| IDMANUF | F07Eh E | 3Fh | Manufacturer identifier register | 0403h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|--|-------------|
| ADDAT2 | F0A0h E | 50h | A/D converter 2 result register | 0000h |
| SSCTB | F0B0h E | 58h | SSC transmit buffer (write-only) | 0000h |
| SSCRB | F0B2h E | 59h | SSC receive buffer (read-only) | XXXXh |
| SSCBR | F0B4h E | 5Ah | SSC baud rate register | 0000h |
| DP0L b | F100h E | 80h | P0L direction control register | -- 00h |
| DP0H b | F102h E | 81h | P0H direction control register | -- 00h |
| DP1L b | F104h E | 82h | P1L direction control register | -- 00h |
| DP1H b | F106h E | 83h | P1H direction control register | -- 00h |
| RP0H b | F108h E | 84h | System start-up configuration register (read-only) | -- XXh |
| CC16IC b | F160h E | B0h | CAPCOM register 16 interrupt control register | -- 00h |
| CC17IC b | F162h E | B1h | CAPCOM register 17 interrupt control register | -- 00h |
| CC18IC b | F164h E | B2h | CAPCOM register 18 interrupt control register | -- 00h |
| CC19IC b | F166h E | B3h | CAPCOM register 19 interrupt control register | -- 00h |
| CC20IC b | F168h E | B4h | CAPCOM register 20 interrupt control register | -- 00h |
| CC21IC b | F16Ah E | B5h | CAPCOM register 21 interrupt control register | -- 00h |
| CC22IC b | F16Ch E | B6h | CAPCOM register 22 interrupt control register | -- 00h |
| CC23IC b | F16Eh E | B7h | CAPCOM register 23 interrupt control register | -- 00h |
| CC24IC b | F170h E | B8h | CAPCOM register 24 interrupt control register | -- 00h |
| CC25IC b | F172h E | B9h | CAPCOM register 25 interrupt control register | -- 00h |
| CC26IC b | F174h E | BAh | CAPCOM register 26 interrupt control register | -- 00h |
| CC27IC b | F176h E | BBh | CAPCOM register 27 interrupt control register | -- 00h |
| CC28IC b | F178h E | BCh | CAPCOM register 28 interrupt control register | -- 00h |
| T7IC b | F17Ah E | BDh | CAPCOM timer 7 interrupt control register | -- 00h |
| T8IC b | F17Ch E | BEh | CAPCOM timer 8 interrupt control register | -- 00h |
| PWMIC b | F17Eh E | BFh | PWM module interrupt control register | -- 00h |
| CC29IC b | F184h E | C2h | CAPCOM register 29 interrupt control register | -- 00h |
| XP0IC b | F186h E | C3h | See Section 8.1 | -- 00h |
| CC30IC b | F18Ch E | C6h | CAPCOM register 30 interrupt control register | -- 00h |
| XP1IC b | F18Eh E | C7h | See Section 8.1 | -- 00h |
| CC31IC b | F194h E | CAh | CAPCOM register 31 interrupt control register | -- 00h |
| XP2IC b | F196h E | CBh | See Section 8.1 | -- 00h |
| S0TBIC b | F19Ch E | CEh | Serial channel 0 transmit buffer interrupt control register. | -- 00h |
| XP3IC b | F19Eh E | CFh | See Section 8.1 | -- 00h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------------|------------------|---------------|---|-------------|
| EXICON b | F1C0h E | E0h | External interrupt control register | 0000h |
| ODP2 b | F1C2h E | E1h | Port2 open drain control register | 0000h |
| PICON b | F1C4h E | E2h | Port input threshold control register | - - 00h |
| ODP3 b | F1C6h E | E3h | Port3 open drain control register | 0000h |
| ODP4 b | F1CAh E | E5h | Port4 open drain control register | - - 00h |
| ODP6 b | F1CEh E | E7h | Port6 open drain control register | - - 00h |
| ODP7 b | F1D2h E | E9h | Port7 open drain control register | - - 00h |
| ODP8 b | F1D6h E | EBh | Port8 open drain control register | - - 00h |
| EXISEL b | F1DAh E | EDh | External interrupt source selection register | 0000h |
| DPP0 | FE00h | 00h | CPU data page pointer 0 register (10-bit) | 0000h |
| DPP1 | FE02h | 01h | CPU data page pointer 1 register (10-bit) | 0001h |
| DPP2 | FE04h | 02h | CPU data page pointer 2 register (10-bit) | 0002h |
| DPP3 | FE06h | 03h | CPU data page pointer 3 register (10-bit) | 0003h |
| CSP | FE08h | 04h | CPU code segment pointer register (read-only) | 0000h |
| EMUCON | FE0Ah | 05h | Emulation control register | - - XXh |
| MDH | FE0Ch | 06h | CPU multiply divide register – High word | 0000h |
| MDL | FE0Eh | 07h | CPU multiply divide register – Low word | 0000h |
| CP | FE10h | 08h | CPU context pointer register | FC00h |
| SP | FE12h | 09h | CPU system stack pointer register | FC00h |
| STKOV | FE14h | 0Ah | CPU stack overflow pointer register | FA00h |
| STKUN | FE16h | 0Bh | CPU stack underflow pointer register | FC00h |
| ADDRSEL1 | FE18h | 0Ch | Address select register 1 | 0000h |
| ADDRSEL2 | FE1Ah | 0Dh | Address select register 2 | 0000h |
| ADDRSEL3 | FE1Ch | 0Eh | Address select register 3 | 0000h |
| ADDRSEL4 | FE1Eh | 0Fh | Address select register 4 | 0000h |
| PW0 | FE30h | 18h | PWM module pulse width register 0 | 0000h |
| PW1 | FE32h | 19h | PWM module pulse width register 1 | 0000h |
| PW2 | FE34h | 1Ah | PWM module pulse width register 2 | 0000h |
| PW3 | FE36h | 1Bh | PWM module pulse width register 3 | 0000h |
| T2 | FE40h | 20h | GPT1 timer 2 register | 0000h |
| T3 | FE42h | 21h | GPT1 timer 3 register | 0000h |
| T4 | FE44h | 22h | GPT1 timer 4 register | 0000h |
| T5 | FE46h | 23h | GPT2 timer 5 register | 0000h |
| T6 | FE48h | 24h | GPT2 timer 6 register | 0000h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|--------|------------------|---------------|----------------------------------|-------------|
| CAPREL | FE4Ah | 25h | GPT2 capture/reload register | 0000h |
| T0 | FE50h | 28h | CAPCOM timer 0 register | 0000h |
| T1 | FE52h | 29h | CAPCOM timer 1 register | 0000h |
| T0REL | FE54h | 2Ah | CAPCOM timer 0 reload register | 0000h |
| T1REL | FE56h | 2Bh | CAPCOM timer 1 reload register | 0000h |
| MAL | FE5Ch | 2Eh | MAC unit accumulator - Low word | 0000h |
| MAH | FE5Eh | 2Fh | MAC unit accumulator - High word | 0000h |
| CC16 | FE60h | 30h | CAPCOM register 16 | 0000h |
| CC17 | FE62h | 31h | CAPCOM register 17 | 0000h |
| CC18 | FE64h | 32h | CAPCOM register 18 | 0000h |
| CC19 | FE66h | 33h | CAPCOM register 19 | 0000h |
| CC20 | FE68h | 34h | CAPCOM register 20 | 0000h |
| CC21 | FE6Ah | 35h | CAPCOM register 21 | 0000h |
| CC22 | FE6Ch | 36h | CAPCOM register 22 | 0000h |
| CC23 | FE6Eh | 37h | CAPCOM register 23 | 0000h |
| CC24 | FE70h | 38h | CAPCOM register 24 | 0000h |
| CC25 | FE72h | 39h | CAPCOM register 25 | 0000h |
| CC26 | FE74h | 3Ah | CAPCOM register 26 | 0000h |
| CC27 | FE76h | 3Bh | CAPCOM register 27 | 0000h |
| CC28 | FE78h | 3Ch | CAPCOM register 28 | 0000h |
| CC29 | FE7Ah | 3Dh | CAPCOM register 29 | 0000h |
| CC30 | FE7Ch | 3Eh | CAPCOM register 30 | 0000h |
| CC31 | FE7Eh | 3Fh | CAPCOM register 31 | 0000h |
| CC0 | FE80h | 40h | CAPCOM register 0 | 0000h |
| CC1 | FE82h | 41h | CAPCOM register 1 | 0000h |
| CC2 | FE84h | 42h | CAPCOM register 2 | 0000h |
| CC3 | FE86h | 43h | CAPCOM register 3 | 0000h |
| CC4 | FE88h | 44h | CAPCOM register 4 | 0000h |
| CC5 | FE8Ah | 45h | CAPCOM register 5 | 0000h |
| CC6 | FE8Ch | 46h | CAPCOM register 6 | 0000h |
| CC7 | FE8Eh | 47h | CAPCOM register 7 | 0000h |
| CC8 | FE90h | 48h | CAPCOM register 8 | 0000h |
| CC9 | FE92h | 49h | CAPCOM register 9 | 0000h |
| CC10 | FE94h | 4Ah | CAPCOM register 10 | 0000h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|-----------|------------------|---------------|--|-------------|
| CC11 | FE96h | 4Bh | CAPCOM register 11 | 0000h |
| CC12 | FE98h | 4Ch | CAPCOM register 12 | 0000h |
| CC13 | FE9Ah | 4Dh | CAPCOM register 13 | 0000h |
| CC14 | FE9Ch | 4Eh | CAPCOM register 14 | 0000h |
| CC15 | FE9Eh | 4Fh | CAPCOM register 15 | 0000h |
| ADDAT | FEA0h | 50h | A/D converter result register | 0000h |
| WDT | FEAEh | 57h | Watchdog timer register (read-only) | 0000h |
| S0TBUF | FEB0h | 58h | Serial channel 0 transmit buffer register (write-only) | 0000h |
| S0RBUF | FEB2h | 59h | Serial channel 0 receive buffer register (read-only) | -- XXh |
| S0BG | FEB4h | 5Ah | Serial channel 0 baud rate generator reload register | 0000h |
| PECC0 | FEC0h | 60h | PEC channel 0 control register | 0000h |
| PECC1 | FEC2h | 61h | PEC channel 1 control register | 0000h |
| PECC2 | FEC4h | 62h | PEC channel 2 control register | 0000h |
| PECC3 | FEC6h | 63h | PEC channel 3 control register | 0000h |
| PECC4 | FEC8h | 64h | PEC channel 4 control register | 0000h |
| PECC5 | FECAh | 65h | PEC channel 5 control register | 0000h |
| PECC6 | FECCh | 66h | PEC channel 6 control register | 0000h |
| PECC7 | FECEh | 67h | PEC channel 7 control register | 0000h |
| P0L b | FF00h | 80h | Port0 low register (lower half of PORT0) | -- 00h |
| P0H b | FF02h | 81h | Port0 high register (upper half of PORT0) | -- 00h |
| P1L b | FF04h | 82h | Port1 low register (lower half of PORT1) | -- 00h |
| P1H b | FF06h | 83h | Port1 high register (upper half of PORT1) | -- 00h |
| IDX0 b | FF08h | 84h | MAC unit address pointer 0 | 0000h |
| IDX1 b | FF0Ah | 85h | MAC unit address pointer 1 | 0000h |
| BUSCON0 b | FF0Ch | 86h | Bus configuration register 0 | 0xx0h |
| MDC b | FF0Eh | 87h | CPU multiply divide control register | 0000h |
| PSW b | FF10h | 88h | CPU program status word | 0000h |
| SYSCON b | FF12h | 89h | CPU system configuration register | 0xx0h |
| BUSCON1 b | FF14h | 8Ah | Bus configuration register 1 | 0000h |
| BUSCON2 b | FF16h | 8Bh | Bus configuration register 2 | 0000h |
| BUSCON3 b | FF18h | 8Ch | Bus configuration register 3 | 0000h |
| BUSCON4 b | FF1Ah | 8Dh | Bus configuration register 4 | 0000h |
| ZEROS b | FF1Ch | 8Eh | Constant value 0's register (read-only) | 0000h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|------------------|------------------|---------------|--|-------------|
| ONES b | FF1Eh | 8Fh | Constant value 1's register (read-only) | FFFFh |
| T78CON b | FF20h | 90h | CAPCOM timer 7 and 8 control register | 0000h |
| CCM4 b | FF22h | 91h | CAPCOM mode control register 4 | 0000h |
| CCM5 b | FF24h | 92h | CAPCOM mode control register 5 | 0000h |
| CCM6 b | FF26h | 93h | CAPCOM mode control register 6 | 0000h |
| CCM7 b | FF28h | 94h | CAPCOM mode control register 7 | 0000h |
| PWMCON0 b | FF30h | 98h | PWM module control register 0 | 0000h |
| PWMCON1 b | FF32h | 99h | PWM module control register 1 | 0000h |
| T2CON b | FF40h | A0h | GPT1 timer 2 control register | 0000h |
| T3CON b | FF42h | A1h | GPT1 timer 3 control register | 0000h |
| T4CON b | FF44h | A2h | GPT1 timer 4 control register | 0000h |
| T5CON b | FF46h | A3h | GPT2 timer 5 control register | 0000h |
| T6CON b | FF48h | A4h | GPT2 timer 6 control register | 0000h |
| T01CON b | FF50h | A8h | CAPCOM timer 0 and timer 1 control register | 0000h |
| CCM0 b | FF52h | A9h | CAPCOM mode control register 0 | 0000h |
| CCM1 b | FF54h | AAh | CAPCOM mode control register 1 | 0000h |
| CCM2 b | FF56h | ABh | CAPCOM mode control register 2 | 0000h |
| CCM3 b | FF58h | ACH | CAPCOM mode control register 3 | 0000h |
| T2IC b | FF60h | B0h | GPT1 timer 2 interrupt control register | -- 00h |
| T3IC b | FF62h | B1h | GPT1 timer 3 interrupt control register | -- 00h |
| T4IC b | FF64h | B2h | GPT1 timer 4 interrupt control register | -- 00h |
| T5IC b | FF66h | B3h | GPT2 timer 5 interrupt control register | -- 00h |
| T6IC b | FF68h | B4h | GPT2 timer 6 interrupt control register | -- 00h |
| CRIC b | FF6Ah | B5h | GPT2 CAPREL interrupt control register | -- 00h |
| S0TIC b | FF6Ch | B6h | Serial channel 0 transmit interrupt control register | -- 00h |
| S0RIC b | FF6Eh | B7h | Serial channel 0 receive interrupt control register | -- 00h |
| S0EIC b | FF70h | B8h | Serial channel 0 error interrupt control register | -- 00h |
| SSCTIC b | FF72h | B9h | SSC transmit interrupt control register | -- 00h |
| SSCRIC b | FF74h | BAh | SSC receive interrupt control register | -- 00h |
| SSCEIC b | FF76h | BBh | SSC error interrupt control register | -- 00h |
| CC0IC b | FF78h | BC h | CAPCOM register 0 interrupt control register | -- 00h |
| CC1IC b | FF7Ah | BDh | CAPCOM register 1 interrupt control register | -- 00h |
| CC2IC b | FF7Ch | BEh | CAPCOM register 2 interrupt control register | -- 00h |
| CC3IC b | FF7Eh | BFh | CAPCOM register 3 interrupt control register | -- 00h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|------------------|------------------|---------------|--|-------------|
| CC4IC b | FF80h | C0h | CAPCOM register 4 interrupt control register | --00h |
| CC5IC b | FF82h | C1h | CAPCOM register 5 interrupt control register | --00h |
| CC6IC b | FF84h | C2h | CAPCOM register 6 interrupt control register | --00h |
| CC7IC b | FF86h | C3h | CAPCOM register 7 interrupt control register | --00h |
| CC8IC b | FF88h | C4h | CAPCOM register 8 interrupt control register | --00h |
| CC9IC b | FF8Ah | C5h | CAPCOM register 9 interrupt control register | --00h |
| CC10IC b | FF8Ch | C6h | CAPCOM register 10 interrupt control register | --00h |
| CC11IC b | FF8Eh | C7h | CAPCOM register 11 interrupt control register | --00h |
| CC12IC b | FF90h | C8h | CAPCOM register 12 interrupt control register | --00h |
| CC13IC b | FF92h | C9h | CAPCOM register 13 interrupt control register | --00h |
| CC14IC b | FF94h | CAh | CAPCOM register 14 interrupt control register | --00h |
| CC15IC b | FF96h | CBh | CAPCOM register 15 interrupt control register | --00h |
| ADCIC b | FF98h | CCh | A/D converter end of conversion interrupt control register | --00h |
| ADEIC b | FF9Ah | CDh | A/D converter overrun error interrupt control register | --00h |
| T0IC b | FF9Ch | CEh | CAPCOM timer 0 interrupt control register | --00h |
| T1IC b | FF9Eh | CFh | CAPCOM timer 1 interrupt control register | --00h |
| ADCON b | FFA0h | D0h | A/D converter control register | 0000h |
| P5 b | FFA2h | D1h | Port 5 register (read-only) | XXXXh |
| P5DIDIS b | FFA4h | D2h | Port 5 digital disable register | 0000h |
| TFR b | FFACh | D6h | Trap flag register | 0000h |
| WDTCON b | FFAEh | D7h | Watchdog timer control register | 00xxh |
| S0CON b | FFB0h | D8h | Serial channel 0 control register | 0000h |
| SSCCON b | FFB2h | D9h | SSC control register | 0000h |
| P2 b | FFC0h | E0h | Port 2 register | 0000h |
| DP2 b | FFC2h | E1h | Port 2 direction control register | 0000h |
| P3 b | FFC4h | E2h | Port 3 register | 0000h |
| DP3 b | FFC6h | E3h | Port 3 direction control register | 0000h |
| P4 b | FFC8h | E4h | Port 4 register (8-bit) | --00h |
| DP4 b | FFCAh | E5h | Port 4 direction control register | --00h |
| P6 b | FFCCh | E6h | Port 6 register (8-bit) | --00h |
| DP6 b | FFCEh | E7h | Port 6 direction control register | --00h |
| P7 b | FFD0h | E8h | Port 7 register (8-bit) | --00h |
| DP7 b | FFD2h | E9h | Port 7 direction control register | --00h |

Table 68. Special function registers ordered by address (continued)

| Name | Physical address | 8-bit address | Description | Reset value |
|--------------|------------------|---------------|-----------------------------------|-------------|
| P8 b | FFD4h | EAh | Port 8 register (8-bit) | --00h |
| DP8 b | FFD6h | EBh | Port 8 direction control register | --00h |
| MRW b | FFDAh | EDh | MAC unit repeat word | 0000h |
| MCW b | FFDCh | EEh | MAC unit control word | 0000h |
| MSW b | FFDEh | EFh | MAC unit status word | 0200h |

22.5 X-registers sorted by name

The following table lists by order of their names all X-Bus registers which are implemented in the ST10F276. Although also physically mapped on X-Bus memory space, the Flash control registers are listed in a separate section, .

Note: The X-registers are not bit-addressable.

Table 69. X-Registers ordered by name

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------|-------------|
| CAN1BRPER | EF0Ch | CAN1: BRP extension register | 0000h |
| CAN1BTR | EF06h | CAN1: Bit timing register | 2301h |
| CAN1CR | EF00h | CAN1: CAN control register | 0001h |
| CAN1EC | EF04h | CAN1: Error counter | 0000h |
| CAN1IF1A1 | EF18h | CAN1: IF1 arbitration 1 | 0000h |
| CAN1IF1A2 | EF1Ah | CAN1: IF1 arbitration 2 | 0000h |
| CAN1IF1CM | EF12h | CAN1: IF1 command mask | 0000h |
| CAN1IF1CR | EF10h | CAN1: IF1 command request | 0001h |
| CAN1IF1DA1 | EF1Eh | CAN1: IF1 data A 1 | 0000h |
| CAN1IF1DA2 | EF20h | CAN1: IF1 data A 2 | 0000h |
| CAN1IF1DB1 | EF22h | CAN1: IF1 data B 1 | 0000h |
| CAN1IF1DB2 | EF24h | CAN1: IF1 data B 2 | 0000h |
| CAN1IF1M1 | EF14h | CAN1: IF1 mask 1 | FFFFh |
| CAN1IF1M2 | EF16h | CAN1: IF1 mask 2 | FFFFh |
| CAN1IF1MC | EF1Ch | CAN1: IF1 message control | 0000h |
| CAN1IF2A1 | EF48h | CAN1: IF2 arbitration 1 | 0000h |
| CAN1IF2A2 | EF4Ah | CAN1: IF2 arbitration 2 | 0000h |
| CAN1IF2CM | EF42h | CAN1: IF2 command mask | 0000h |
| CAN1IF2CR | EF40h | CAN1: IF2 command request | 0001h |
| CAN1IF2DA1 | EF4Eh | CAN1: IF2 data A 1 | 0000h |

Table 69. X-Registers ordered by name (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------|-------------|
| CAN1IF2DA2 | EF50h | CAN1: IF2 data A 2 | 0000h |
| CAN1IF2DB1 | EF52h | CAN1: IF2 data B 1 | 0000h |
| CAN1IF2DB2 | EF54h | CAN1: IF2 data B 2 | 0000h |
| CAN1IF2M1 | EF44h | CAN1: IF2 mask 1 | FFFFh |
| CAN1IF2M2 | EF46h | CAN1: IF2 mask 2 | FFFFh |
| CAN1IF2MC | EF4Ch | CAN1: IF2 message control | 0000h |
| CAN1IP1 | EFA0h | CAN1: interrupt pending 1 | 0000h |
| CAN1IP2 | EFA2h | CAN1: interrupt pending 2 | 0000h |
| CAN1IR | EF08h | CAN1: interrupt register | 0000h |
| CAN1MV1 | EFB0h | CAN1: Message valid 1 | 0000h |
| CAN1MV2 | EFB2h | CAN1: Message valid 2 | 0000h |
| CAN1ND1 | EF90h | CAN1: New data 1 | 0000h |
| CAN1ND2 | EF92h | CAN1: New data 2 | 0000h |
| CAN1SR | EF02h | CAN1: Status register | 0000h |
| CAN1TR | EF0Ah | CAN1: Test register | 00x0h |
| CAN1TR1 | EF80h | CAN1: Transmission request 1 | 0000h |
| CAN1TR2 | EF82h | CAN1: Transmission request 2 | 0000h |
| CAN2BRPER | EE0Ch | CAN2: BRP extension register | 0000h |
| CAN2BTR | EE06h | CAN2: Bit timing register | 2301h |
| CAN2CR | EE00h | CAN2: CAN control register | 0001h |
| CAN2EC | EE04h | CAN2: Error counter | 0000h |
| CAN2IF1A1 | EE18h | CAN2: IF1 arbitration 1 | 0000h |
| CAN2IF1A2 | EE1Ah | CAN2: IF1 arbitration 2 | 0000h |
| CAN2IF1CM | EE12h | CAN2: IF1 command mask | 0000h |
| CAN2IF1CR | EE10h | CAN2: IF1 command request | 0001h |
| CAN2IF1DA1 | EE1Eh | CAN2: IF1 data A 1 | 0000h |
| CAN2IF1DA2 | EE20h | CAN2: IF1 data A 2 | 0000h |
| CAN2IF1DB1 | EE22h | CAN2: IF1 data B 1 | 0000h |
| CAN2IF1DB2 | EE24h | CAN2: IF1 data B 2 | 0000h |
| CAN2IF1M1 | EE14h | CAN2: IF1 mask 1 | FFFFh |
| CAN2IF1M2 | EE16h | CAN2: IF1 mask 2 | FFFFh |
| CAN2IF1MC | EE1Ch | CAN2: IF1 message control | 0000h |
| CAN2IF2A1 | EE48h | CAN2: IF2 arbitration 1 | 0000h |
| CAN2IF2A2 | EE4Ah | CAN2: IF2 arbitration 2 | 0000h |

Table 69. X-Registers ordered by name (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------------|-------------|
| CAN2IF2CM | EE42h | CAN2: IF2 command mask | 0000h |
| CAN2IF2CR | EE40h | CAN2: IF2 command request | 0001h |
| CAN2IF2DA1 | EE4Eh | CAN2: IF2 data A 1 | 0000h |
| CAN2IF2DA2 | EE50h | CAN2: IF2 data A 2 | 0000h |
| CAN2IF2DB1 | EE52h | CAN2: IF2 data B 1 | 0000h |
| CAN2IF2DB2 | EE54h | CAN2: IF2 data B 2 | 0000h |
| CAN2IF2M1 | EE44h | CAN2: IF2 mask 1 | FFFFh |
| CAN2IF2M2 | EE46h | CAN2: IF2 mask 2 | FFFFh |
| CAN2IF2MC | EE4Ch | CAN2: IF2 message control | 0000h |
| CAN2IP1 | EEA0h | CAN2: Interrupt pending 1 | 0000h |
| CAN2IP2 | EEA2h | CAN2: Interrupt pending 2 | 0000h |
| CAN2IR | EE08h | CAN2: Interrupt register | 0000h |
| CAN2MV1 | EEB0h | CAN2: Message valid 1 | 0000h |
| CAN2MV2 | EEB2h | CAN2: Message valid 2 | 0000h |
| CAN2ND1 | EE90h | CAN2: New data 1 | 0000h |
| CAN2ND2 | EE92h | CAN2: New data 2 | 0000h |
| CAN2SR | EE02h | CAN2: Status register | 0000h |
| CAN2TR | EE0Ah | CAN2: Test register | 00x0h |
| CAN2TR1 | EE80h | CAN2: Transmission request 1 | 0000h |
| CAN2TR2 | EE82h | CAN2: Transmission request 2 | 0000h |
| I2CCCR1 | EA06h | I2C Clock control register 1 | 0000h |
| I2CCCR2 | EA0Eh | I2C Clock control register 2 | 0000h |
| I2CCR | EA00h | I2C Control register | 0000h |
| I2CDR | EA0Ch | I2C Data register | 0000h |
| I2COAR1 | EA08h | I2C Own address register 1 | 0000h |
| I2COAR2 | EA0Ah | I2C Own address register 2 | 0000h |
| I2CSR1 | EA02h | I2C Status register 1 | 0000h |
| I2CSR2 | EA04h | I2C Status register 2 | 0000h |
| RTCAH | ED14h | RTC Alarm register high byte | XXXXh |
| RTCAL | ED12h | RTC Alarm register low byte | XXXXh |
| RTCCON | ED00h | RTC Control register | 000Xh |
| RTCDH | ED0Ch | RTC Divider counter high byte | XXXXh |
| RTCDL | ED0Ah | RTC Divider counter low byte | XXXXh |
| RTCH | ED10h | RTC Programmable counter high byte | XXXXh |

Table 69. X-Registers ordered by name (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|---|-------------|
| RTCL | ED0Eh | RTC Programmable counter low byte | XXXXh |
| RTCPH | ED08h | RTC Prescaler register high byte | XXXXh |
| RTCPL | ED06h | RTC Prescaler register low byte | XXXXh |
| XCLKOUTDIV | EB02h | CLKOUT Divider control register | - - 00h |
| XEMU0 | EB76h | XBUS Emulation register 0 (write-only) | XXXXh |
| XEMU1 | EB78h | XBUS Emulation register 1 (write-only) | XXXXh |
| XEMU2 | EB7Ah | XBUS Emulation register 2 (write-only) | XXXXh |
| XEMU3 | EB7Ch | XBUS Emulation register 3 (write-only) | XXXXh |
| XIR0CLR | EB14h | X-Interrupt 0 clear register (write-only) | 0000h |
| XIR0SEL | EB10h | X-Interrupt 0 selection register | 0000h |
| XIR0SET | EB12h | X-Interrupt 0 set register (write-only) | 0000h |
| XIR1CLR | EB24h | X-Interrupt 1 clear register (write-only) | 0000h |
| XIR1SEL | EB20h | X-Interrupt 1 selection register | 0000h |
| XIR1SET | EB22h | X-Interrupt 1 set register (write-only) | 0000h |
| XIR2CLR | EB34h | X-Interrupt 2 clear register (write-only) | 0000h |
| XIR2SEL | EB30h | X-Interrupt 2 selection register | 0000h |
| XIR2SET | EB32h | X-Interrupt 2 set register (write-only) | 0000h |
| XIR3CLR | EB44h | X-Interrupt 3 clear selection register (write-only) | 0000h |
| XIR3SEL | EB40h | X-Interrupt 3 selection register | 0000h |
| XIR3SET | EB42h | X-Interrupt 3 set selection register (write-only) | 0000h |
| XMISC | EB46h | XBUS miscellaneous features register | 0000h |
| XP1DIDIS | EB36h | Port 1 digital disable register | 0000h |
| XPEREMU | EB7Eh | XPERCON copy for emulation (write-only) | XXXXh |
| XPICON | EB26h | Extended port input threshold control register | - - 00h |
| XPOLAR | EC04h | XPWM module channel polarity register | 0000h |
| XPP0 | EC20h | XPWM module period register 0 | 0000h |
| XPP1 | EC22h | XPWM module period register 1 | 0000h |
| XPP2 | EC24h | XPWM module period register 2 | 0000h |
| XPP3 | EC26h | XPWM module period register 3 | 0000h |
| XPT0 | EC10h | XPWM module up/down counter 0 | 0000h |
| XPT1 | EC12h | XPWM module up/down counter 1 | 0000h |
| XPT2 | EC14h | XPWM module up/down counter 2 | 0000h |

Table 69. X-Registers ordered by name (continued)

| Name | Physical address | Description | Reset value |
|-------------|------------------|---|-------------|
| XPT3 | EC16h | XPWM module up/down counter 3 | 0000h |
| XPW0 | EC30h | XPWM module pulse width register 0 | 0000h |
| XPW1 | EC32h | XPWM module pulse width register 1 | 0000h |
| XPW2 | EC34h | XPWM module pulse width register 2 | 0000h |
| XPW3 | EC36h | XPWM module pulse width register 3 | 0000h |
| XPWMCON0 | EC00h | XPWM module control register 0 | 0000h |
| XPWMCON0CLR | EC08h | XPWM module clear control reg. 0 (write-only) | 0000h |
| XPWMCON0SET | EC06h | XPWM module set control register 0 (write-only) | 0000h |
| XPWMCON1 | EC02h | XPWM module control register 1 | 0000h |
| XPWMCON1CLR | EC0Ch | XPWM module clear control reg. 0 (write-only) | 0000h |
| XPWMCON1SET | EC0Ah | XPWM module set control register 0 (write-only) | 0000h |
| XPWMPORT | EC80h | XPWM module port control register | 0000h |
| XS1BG | E906h | XASC baud rate generator reload register | 0000h |
| XS1CON | E900h | XASC control register | 0000h |
| XS1CONCLR | E904h | XASC clear control register (write-only) | 0000h |
| XS1CONSET | E902h | XASC set control register (write-only) | 0000h |
| XS1PORT | E980h | XASC port control register | 0000h |
| XS1RBUF | E90Ah | XASC receive buffer register | 0000h |
| XS1TBUF | E908h | XASC transmit buffer register | 0000h |
| XSSCBR | E80Ah | XSSC baud rate register | 0000h |
| XSSCCON | E800h | XSSC control register | 0000h |
| XSSCCONCLR | E804h | XSSC clear control register (write-only) | 0000h |
| XSSCCONSET | E802h | XSSC set control register (write-only) | 0000h |
| XSSCPORT | E880h | XSSC port control register | 0000h |
| XSSCRB | E808h | XSSC receive buffer | XXXXh |
| XSSCTB | E806h | XSSC transmit buffer | 0000h |

22.6 X-registers ordered by address

The following table lists by order of their physical addresses all X-Bus registers which are implemented in the ST10F276. Although also physically mapped on X-Bus memory space, the Flash control registers are listed in a separate section, .

Note: The X-registers are not bit-addressable.

Table 70. X-registers ordered by address

| Name | Physical address | Description | Reset value |
|------------|------------------|---|-------------|
| XSSCCON | E800h | XSSC control register | 0000h |
| XSSCCONSET | E802h | XSSC set control register (write-only) | 0000h |
| XSSCCONCLR | E804h | XSSC clear control register (write-only) | 0000h |
| XSSCTB | E806h | XSSC transmit buffer | 0000h |
| XSSCRB | E808h | XSSC receive buffer | XXXXh |
| XSSCBR | E80Ah | XSSC baud rate register | 0000h |
| XSSCPORT | E880h | XSSC port control register | 0000h |
| XS1CON | E900h | XASC control register | 0000h |
| XS1CONSET | E902h | XASC set control register (write-only) | 0000h |
| XS1CONCLR | E904h | XASC clear control register (write-only) | 0000h |
| XS1BG | E906h | XASC baud rate generator reload register | 0000h |
| XS1TBUF | E908h | XASC transmit buffer register | 0000h |
| XS1RBUF | E90Ah | XASC receive buffer register | 0000h |
| XS1PORT | E980h | XASC port control register | 0000h |
| I2CCR | EA00h | I2C control register | 0000h |
| I2CSR1 | EA02h | I2C status register 1 | 0000h |
| I2CSR2 | EA04h | I2C status register 2 | 0000h |
| I2CCCR1 | EA06h | I2C clock control register 1 | 0000h |
| I2COAR1 | EA08h | I2C own address register 1 | 0000h |
| I2COAR2 | EA0Ah | I2C own address register 2 | 0000h |
| I2CDR | EA0Ch | I2C data register | 0000h |
| I2CCCR2 | EA0Eh | I2C clock control register 2 | 0000h |
| XCLKOUTDIV | EB02h | CLKOUT divider control register | - - 00h |
| XIR0SEL | EB10h | X-Interrupt 0 selection register | 0000h |
| XIR0SET | EB12h | X-Interrupt 0 set register (write-only) | 0000h |
| XIR0CLR | EB14h | X-Interrupt 0 clear register (write-only) | 0000h |
| XIR1SEL | EB20h | X-Interrupt 1 selection register | 0000h |
| XIR1SET | EB22h | X-Interrupt 1 set register (write-only) | 0000h |
| XIR1CLR | EB24h | X-Interrupt 1 clear register (write-only) | 0000h |

Table 70. X-registers ordered by address (continued)

| Name | Physical address | Description | Reset value |
|-------------|------------------|---|-------------|
| XPICON | EB26h | Extended port input threshold control register | - - 00h |
| XIR2SEL | EB30h | X-Interrupt 2 selection register | 0000h |
| XIR2SET | EB32h | X-Interrupt 2 set register (write-only) | 0000h |
| XIR2CLR | EB34h | X-Interrupt 2 clear register (write-only) | 0000h |
| XP1DIDIS | EB36h | Port 1 digital disable register | 0000h |
| XIR3SEL | EB40h | X-Interrupt 3 selection register | 0000h |
| XIR3SET | EB42h | X-Interrupt 3 set selection register (write-only) | 0000h |
| XIR3CLR | EB44h | X-Interrupt 3 clear selection register (write-only) | 0000h |
| XMISC | EB46h | XBUS miscellaneous features register | 0000h |
| XEMU0 | EB76h | XBUS emulation register 0 (write-only) | XXXXh |
| XEMU1 | EB78h | XBUS emulation register 1 (write-only) | XXXXh |
| XEMU2 | EB7Ah | XBUS emulation register 2 (write-only) | XXXXh |
| XEMU3 | EB7Ch | XBUS emulation register 3 (write-only) | XXXXh |
| XPEREMU | EB7Eh | XPICON copy for emulation (write-only) | XXXXh |
| XPWMCON0 | EC00h | XPWM module control register 0 | 0000h |
| XPWMCON1 | EC02h | XPWM module control register 1 | 0000h |
| XPOLAR | EC04h | XPWM module channel polarity register | 0000h |
| XPWMCON0SET | EC06h | XPWM module set control register 0 (write-only) | 0000h |
| XPWMCON0CLR | EC08h | XPWM module clear control reg. 0 (write-only) | 0000h |
| XPWMCON1SET | EC0Ah | XPWM module set control register 0 (write-only) | 0000h |
| XPWMCON1CLR | EC0Ch | XPWM module clear control reg. 0 (write-only) | 0000h |
| XPT0 | EC10h | XPWM module up/down counter 0 | 0000h |
| XPT1 | EC12h | XPWM module up/down counter 1 | 0000h |
| XPT2 | EC14h | XPWM module up/down Counter 2 | 0000h |
| XPT3 | EC16h | XPWM module up/down counter 3 | 0000h |
| XPP0 | EC20h | XPWM module period register 0 | 0000h |
| XPP1 | EC22h | XPWM module period register 1 | 0000h |
| XPP2 | EC24h | XPWM module period register 2 | 0000h |
| XPP3 | EC26h | XPWM module period register 3 | 0000h |
| XPW0 | EC30h | XPWM module pulse width register 0 | 0000h |

Table 70. X-registers ordered by address (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------------|-------------|
| XPW1 | EC32h | XPWM module pulse width register 1 | 0000h |
| XPW2 | EC34h | XPWM module pulse width register 2 | 0000h |
| XPW3 | EC36h | XPWM module pulse width register 3 | 0000h |
| XPWMPORT | EC80h | XPWM module port control register | 0000h |
| RTCCON | ED00h | RTC control register | 000Xh |
| RTCPL | ED06h | RTC prescaler register low byte | XXXXh |
| RTCPH | ED08h | RTC prescaler register high byte | XXXXh |
| RTCDL | ED0Ah | RTC divider counter low byte | XXXXh |
| RTCDH | ED0Ch | RTC divider counter high byte | XXXXh |
| RTCL | ED0Eh | RTC programmable counter low byte | XXXXh |
| RTCH | ED10h | RTC programmable counter high byte | XXXXh |
| RTCAL | ED12h | RTC alarm register low byte | XXXXh |
| RTCAH | ED14h | RTC alarm register high byte | XXXXh |
| CAN2CR | EE00h | CAN2: CAN control register | 0001h |
| CAN2SR | EE02h | CAN2: status register | 0000h |
| CAN2EC | EE04h | CAN2: error counter | 0000h |
| CAN2BTR | EE06h | CAN2: bit timing register | 2301h |
| CAN2IR | EE08h | CAN2: interrupt register | 0000h |
| CAN2TR | EE0Ah | CAN2: test register | 00x0h |
| CAN2BRPER | EE0Ch | CAN2: BRP extension register | 0000h |
| CAN2IF1CR | EE10h | CAN2: IF1 command request | 0001h |
| CAN2IF1CM | EE12h | CAN2: IF1 command mask | 0000h |
| CAN2IF1M1 | EE14h | CAN2: IF1 mask 1 | FFFFh |
| CAN2IF1M2 | EE16h | CAN2: IF1 mask 2 | FFFFh |
| CAN2IF1A1 | EE18h | CAN2: IF1 arbitration 1 | 0000h |
| CAN2IF1A2 | EE1Ah | CAN2: IF1 arbitration 2 | 0000h |
| CAN2IF1MC | EE1Ch | CAN2: IF1 message control | 0000h |
| CAN2IF1DA1 | EE1Eh | CAN2: IF1 data A 1 | 0000h |
| CAN2IF1DA2 | EE20h | CAN2: IF1 data A 2 | 0000h |
| CAN2IF1DB1 | EE22h | CAN2: IF1 data B 1 | 0000h |
| CAN2IF1DB2 | EE24h | CAN2: IF1 data B 2 | 0000h |
| CAN2IF2CR | EE40h | CAN2: IF2 command request | 0001h |
| CAN2IF2CM | EE42h | CAN2: IF2 command mask | 0000h |
| CAN2IF2M1 | EE44h | CAN2: IF2 mask 1 | FFFFh |

Table 70. X-registers ordered by address (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------|-------------|
| CAN2IF2M2 | EE46h | CAN2: IF2 mask 2 | FFFFh |
| CAN2IF2A1 | EE48h | CAN2: IF2 arbitration 1 | 0000h |
| CAN2IF2A2 | EE4Ah | CAN2: IF2 arbitration 2 | 0000h |
| CAN2IF2MC | EE4Ch | CAN2: IF2 message control | 0000h |
| CAN2IF2DA1 | EE4Eh | CAN2: IF2 data A 1 | 0000h |
| CAN2IF2DA2 | EE50h | CAN2: IF2 data A 2 | 0000h |
| CAN2IF2DB1 | EE52h | CAN2: IF2 data B 1 | 0000h |
| CAN2IF2DB2 | EE54h | CAN2: IF2 data B 2 | 0000h |
| CAN2TR1 | EE80h | CAN2: transmission request 1 | 0000h |
| CAN2TR2 | EE82h | CAN2: transmission request 2 | 0000h |
| CAN2ND1 | EE90h | CAN2: new data 1 | 0000h |
| CAN2ND2 | EE92h | CAN2: new data 2 | 0000h |
| CAN2IP1 | EEA0h | CAN2: interrupt pending 1 | 0000h |
| CAN2IP2 | EEA2h | CAN2: interrupt pending 2 | 0000h |
| CAN2MV1 | EEB0h | CAN2: message valid 1 | 0000h |
| CAN2MV2 | EEB2h | CAN2: message valid 2 | 0000h |
| CAN1CR | EF00h | CAN1: CAN control register | 0001h |
| CAN1SR | EF02h | CAN1: status register | 0000h |
| CAN1EC | EF04h | CAN1: error counter | 0000h |
| CAN1BTR | EF06h | CAN1: bit timing register | 2301h |
| CAN1IR | EF08h | CAN1: interrupt register | 0000h |
| CAN1TR | EF0Ah | CAN1: test register | 00x0h |
| CAN1BRPER | EF0Ch | CAN1: BRP extension register | 0000h |
| CAN1IF1CR | EF10h | CAN1: IF1 command request | 0001h |
| CAN1IF1CM | EF12h | CAN1: IF1 command mask | 0000h |
| CAN1IF1M1 | EF14h | CAN1: IF1 mask 1 | FFFFh |
| CAN1IF1M2 | EF16h | CAN1: IF1 mask 2 | FFFFh |
| CAN1IF1A1 | EF18h | CAN1: IF1 arbitration 1 | 0000h |
| CAN1IF1A2 | EF1Ah | CAN1: IF1 arbitration 2 | 0000h |
| CAN1IF1MC | EF1Ch | CAN1: IF1 message control | 0000h |
| CAN1IF1DA1 | EF1Eh | CAN1: IF1 data A 1 | 0000h |
| CAN1IF1DA2 | EF20h | CAN1: IF1 data A 2 | 0000h |
| CAN1IF1DB1 | EF22h | CAN1: IF1 data B 1 | 0000h |
| CAN1IF1DB2 | EF24h | CAN1: IF1 data B 2 | 0000h |

Table 70. X-registers ordered by address (continued)

| Name | Physical address | Description | Reset value |
|------------|------------------|------------------------------|-------------|
| CAN1IF2CR | EF40h | CAN1: IF2 command request | 0001h |
| CAN1IF2CM | EF42h | CAN1: IF2 command mask | 0000h |
| CAN1IF2M1 | EF44h | CAN1: IF2 mask 1 | FFFFh |
| CAN1IF2M2 | EF46h | CAN1: IF2 mask 2 | FFFFh |
| CAN1IF2A1 | EF48h | CAN1: IF2 arbitration 1 | 0000h |
| CAN1IF2A2 | EF4Ah | CAN1: IF2 arbitration 2 | 0000h |
| CAN1IF2MC | EF4Ch | CAN1: IF2 message control | 0000h |
| CAN1IF2DA1 | EF4Eh | CAN1: IF2 data A 1 | 0000h |
| CAN1IF2DA2 | EF50h | CAN1: IF2 data A 2 | 0000h |
| CAN1IF2DB1 | EF52h | CAN1: IF2 data B 1 | 0000h |
| CAN1IF2DB2 | EF54h | CAN1: IF2 data B 2 | 0000h |
| CAN1TR1 | EF80h | CAN1: transmission request 1 | 0000h |
| CAN1TR2 | EF82h | CAN1: transmission request 2 | 0000h |
| CAN1ND1 | EF90h | CAN1: new data 1 | 0000h |
| CAN1ND2 | EF92h | CAN1: new data 2 | 0000h |
| CAN1IP1 | EFA0h | CAN1: interrupt pending 1 | 0000h |
| CAN1IP2 | EFA2h | CAN1: interrupt pending 2 | 0000h |
| CAN1MV1 | EFB0h | CAN1: message valid 1 | 0000h |
| CAN1MV2 | EFB2h | CAN1: message valid 2 | 0000h |

22.7 Flash registers ordered by name

The following table lists by order of their names all FLASH control registers which are implemented in the ST10F276. Note that as they are physically mapped on the X-Bus, these registers are not bit-addressable.

Table 71. Flash registers ordered by name

| Name | Physical address | Description | Reset value |
|----------|------------------|---|-------------|
| FARH | 0x000E 0012 | Flash address register High | 0000h |
| FARL | 0x000E 0010 | Flash address register Low | 0000h |
| FCR0H | 0x000E 0002 | Flash control register 0 - High | 0000h |
| FCR0L | 0x000E 0000 | Flash control register 0 - Low | 0000h |
| FCR1H | 0x000E 0006 | Flash control register 1 - High | 0000h |
| FCR1L | 0x000E 0004 | Flash control register 1 - Low | 0000h |
| FDR0H | 0x000E 000A | Flash data register 0 - High | FFFFh |
| FDR0L | 0x000E 0008 | Flash data register 0 - Low | FFFFh |
| FDR1H | 0x000E 000E | Flash data register 1 - High | FFFFh |
| FDR1L | 0x000E 000C | Flash data register 1 - Low | FFFFh |
| FER | 0x000E 0014 | Flash error register | 0000h |
| FNVAPR0 | 0x000E DFB8 | Flash nonvolatile access protection Reg. 0 | ACFFh |
| FNVAPR1H | 0x000E DFBE | Flash nonvolatile access protection Reg. 1 - High | FFFFh |
| FNVAPR1L | 0x000E DFBC | Flash nonvolatile access protection Reg. 1 - Low | FFFFh |
| FNVWPIRH | 0x000E DFB6 | Flash nonvolatile protection I Reg. High | FFFFh |
| FNVWPIRL | 0x000E DFB4 | Flash nonvolatile protection I Reg. Low | FFFFh |
| FNVWPXRH | 0x000E DFB2 | Flash nonvolatile protection X Reg. High | FFFFh |
| FNVWPXRL | 0x000E DFB0 | Flash nonvolatile protection X Reg. Low | FFFFh |
| XFICR | 0x000E E000 | XFlash interface control register | 000Fh |

22.8 Flash registers ordered by address

The following table lists by order of their physical addresses all FLASH control registers which are implemented in the ST10F276. Note that as they are physically mapped on the X-Bus, these registers are not bit-addressable.

Table 72. FLASH registers ordered by address

| Name | Physical address | Description | Reset value |
|----------|------------------|---|-------------|
| FCR0L | 0x000E 0000 | Flash control register 0 - Low | 0000h |
| FCR0H | 0x000E 0002 | Flash control register 0 - High | 0000h |
| FCR1L | 0x000E 0004 | Flash control register 1 - Low | 0000h |
| FCR1H | 0x000E 0006 | Flash control register 1 - High | 0000h |
| FDR0L | 0x000E 0008 | Flash data register 0 - Low | FFFFh |
| FDR0H | 0x000E 000A | Flash data register 0 - High | FFFFh |
| FDR1L | 0x000E 000C | Flash data register 1 - Low | FFFFh |
| FDR1H | 0x000E 000E | Flash data register 1 - High | FFFFh |
| FARL | 0x000E 0010 | Flash address register Low | 0000h |
| FARH | 0x000E 0012 | Flash address register High | 0000h |
| FER | 0x000E 0014 | Flash error register | 0000h |
| FNWXPXRL | 0x000E DFB0 | Flash nonvolatile protection X reg. Low | FFFFh |
| FNWXPXRH | 0x000E DFB2 | Flash nonvolatile protection X reg. High | FFFFh |
| FNWPIRL | 0x000E DFB4 | Flash nonvolatile protection I reg. Low | FFFFh |
| FNWPIRH | 0x000E DFB6 | Flash nonvolatile protection I reg. High | FFFFh |
| FNVAPR0 | 0x000E DFB8 | Flash nonvolatile access protection reg. 0 | ACFFh |
| FNVAPR1L | 0x000E DFBC | Flash nonvolatile access protection reg. 1 - Low | FFFFh |
| FNVAPR1H | 0x000E DFBE | Flash nonvolatile access protection reg. 1 - High | FFFFh |
| XFICR | 0x000E E000 | XFlash interface control register | 000Fh |

22.9 Identification registers

The ST10F276 have four Identification registers, mapped in ESFR space. These registers contain:

- the manufacturer identifier
- the chip identifier with revision number
- the internal Flash and size identifier
- the programming voltage description

| IDMANUF (F07Eh / 3Fh) | | | | | | | | ESFR | | | | | | | | Reset value:0403h | | | | | | | |
|-----------------------|----|----|----|----|----|---|---|------|---|---|---|---|---|---|---|-------------------|--|--|--|--|--|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | |
| MANUF | | | | | | | | | | | 0 | 0 | 0 | 1 | 1 | | | | | | | | |
| R | | | | | | | | | | | | | | | | | | | | | | | |

Table 73. MANUF description

| Bit | Function |
|-------|---|
| MANUF | Manufacturer identifier 020h: STMicroelectronics manufacturer (JTAG worldwide normalization) |

| IDCHIP (F07Ch / 3Eh) | | | | | | | | ESFR | | | | | | | | Reset value:114xh | | | | | | | |
|----------------------|----|----|----|----|----|---|---|------|---|---|-------|---|---|---|---|-------------------|--|--|--|--|--|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | |
| IDCHIP | | | | | | | | | | | REVID | | | | | | | | | | | | |
| R | | | | | | | | | | | R | | | | | | | | | | | | |

Table 74. IDCHIP description

| Bit | Function |
|--------|--|
| IDCHIP | Device identifier 114h: ST10F276 Identifier (276) |
| REVID | Device revision identifier Xh: According to revision number |

| IDMEM (F07Ah / 3Dh) | | | | | | | | ESFR | | | | | | | | Reset value:30D0h | | | | | | | |
|---------------------|----|----|----|---------|----|---|---|------|---|---|---|---|---|---|---|-------------------|--|--|--|--|--|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | |
| MEMTYP | | | | MEMSIZE | | | | | | | | | | | | | | | | | | | |
| R | | | | R | | | | | | | | | | | | | | | | | | | |

Table 75. IDMEM description

| Bit | Function |
|---------|---|
| MEMSIZE | Internal memory size Internal Memory size is 4 x (MEMSIZE) (in Kbyte) 0D0h for ST10F276 (832 Kbytes) |
| MENTYP | Internal memory type '0h': ROM-Less '1h': (M) ROM memory '2h': (S) Standard FLASH memory '3h': (H) High Performance FLASH memory (ST10F276) '4h...Fh': <i>reserved</i> |

| IDPROG (F078h / 3Ch) | | | | | | | | ESFR | | | | Reset value:0040h | | | |
|----------------------|----|----|----|----|----|---|---|---------|---|---|---|-------------------|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PROGVPP | | | | | | | | PROGVDD | | | | | | | |
| R | | | | | | | | R | | | | | | | |

Table 76. IDPROG description

| Bit | Function |
|---------|--|
| PROGVDD | Programming VDD voltage VDD voltage when programming EPROM or FLASH devices is calculated using the following formula: $VDD = 20 \times [PROGVDD] / 256$ (volts) - 40h for ST10F276 (5V). |
| PROGVPP | Programming VPP voltage (no need of external VPP) - 00h |

Note: All identification words are read-only registers.

The values written inside different Identification Register bits are valid only after the Flash initialization phase is completed. When code execution is started from internal memory (pin EA held high during reset), the Flash has completed its initialization, so the bits of Identification Registers are immediately ready to be read out. On the contrary, when code execution is started from external memory (pin EA held low during reset), the Flash initialization is not yet completed, so the bits of Identification Registers are not ready. The user can poll bits 15 and 14 of IDMEM register: When both bits are read low, the Flash initialization is complete, so all Identification Register bits are correct.

Before Flash initialization completion, the default setting of the different identification registers are the following:

| | |
|---------|------------------------------|
| IDMANUF | 0403h |
| IDCHIP | 114xh (x = silicon revision) |
| IDMEM | F0D0h |
| IDPROG | 0040h |

22.10 System configuration registers

The ST10F276 has registers used for a different configuration of the overall system. These registers are described below.

| SYSCON (FF12h / 89h) | | | | | | SFR | | | | | | Reset value: 0xx0h | | | |
|----------------------|--------|---------|--------|---------|--------|--------|--------|---------|---------|----------|------|--------------------|------|----------|------------|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| STKSZ | ROM S1 | SGT DIS | ROM EN | BYT DIS | CLK EN | WR CFG | CS CFG | PWD CFG | OWD DIS | BDR STEN | XPEN | VISI BLE | XPEN | VISI BLE | XPEN-SHARE |
| RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW |

Note: SYSCON Reset Value is: 0000 0xx0 0x00 0000b

Table 77. SYSCON description

| Bit | Function |
|------------|---|
| XPEN-SHARE | XBUS peripheral share mode control '0': External accesses to XBUS peripherals are disabled. '1': XRAM1 and XRAM2 are accessible via the external bus during hold mode. External accesses to the other XBUS peripherals are not guaranteed in terms of AC timings. |
| VISIBLE | Visible mode control '0': Accesses to XBUS peripherals are done internally. '1': XBUS peripheral accesses are made visible on the external pins. |
| XPEN | XBUS peripheral enable bit '0': Accesses to the on-chip X-peripherals and XRAM are disabled. '1': The on-chip X-peripherals are enabled. |
| BDRSTEN | Bidirectional reset enable '0': RSTIN pin is an input pin only. SW Reset or WDT Reset have no effect on this pin. '1': RSTIN pin is a bidirectional pin. This pin is pulled low during internal reset sequence. |
| OWDDIS | Oscillator watchdog disable control '0': Oscillator Watchdog (OWD) is enabled. If PLL is bypassed, the OWD monitors XTAL1 activity. If there is no activity on XTAL1 for at least 1 µs, the CPU clock is switched automatically to PLL's base frequency (from 750 kHz to 3 MHz). '1': OWD is disabled. If the PLL is bypassed, the CPU clock is always driven by XTAL1 signal. The PLL is turned off to reduce power supply current. |
| PWDCFG | Power down mode configuration control '0': Power Down Mode can only be entered during PWRDN instruction execution if NMI pin is low, otherwise the instruction has no effect. To exit Power Down Mode, an external reset must occur by asserting the RSTIN pin. '1': Power Down Mode can only be entered during PWRDN instruction execution if all enabled fast external interrupt EXxIN pins are in their inactive level. Exiting this mode can be done by asserting one enabled EXxIN pin or with external reset. |
| CSCFG | Chip select configuration control '0': Latched Chip Select lines, CSx changes 1 TCL after rising edge of ALE. '1': Unlatched Chip Select lines, CSx changes with rising edge of ALE. |

Table 77. SYSCON description (continued)

| Bit | Function |
|--------|--|
| WRCFG | Write configuration control (inverted copy of WRC bit of RP0H) '0': Pins WR and BHE retain their normal function. '1': Pin WR acts as WRL, pin BHE acts as WRH. |
| CLKEN | System clock output enable (CLKOUT) '0': CLKOUT disabled, pin may be used for general purpose I/O. '1': CLKOUT enabled, pin outputs the system clock signal or a prescaled value of system clock according to XCLKOUTDIV register setting. |
| BYTDIS | Disable/enable control for pin BHE (set according to data bus width) '0': Pin BHE enabled. '1': Pin BHE disabled, pin may be used for general purpose I/O. |
| ROMEN | Internal memory enable (set according to pin EA during reset) '0': Internal memory disabled: Accesses to the IFlash Memory area use the external bus. '1': Internal memory enabled. |
| SGTDIS | Segmentation disable/enable control '0': Segmentation enabled (CSP is saved/restored during interrupt entry/exit). '1': Segmentation disabled (Only IP is saved/restored). |
| ROMS1 | Internal memory mapping '0': Internal memory area mapped to segment 0 (00'0000h...00'7FFFh). '1': Internal memory area mapped to segment 1 (01'0000h...01'7FFFh). |
| STKSZ | System stack size Selects the size of the system stack (in the internal I-RAM) from 32 to 1024 words. |

BUSCON0 (FF0Ch / 86h) SFR Reset value: 0xx0h

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|---------|--------|----|---------|---------|---|------|-------|-------|------|---|---|---|---|
| CSWEN0 | CSREN0 | RDYPOLO | RDYEN0 | - | BUSACT0 | ALECTL0 | - | BTYP | MTTC0 | RWDC0 | MCTC | | | | |
| RW | RW | | RW | | RW | RW | | RW | RW | RW | RW | | | | |

BUSCON1 (FF14h / 8Ah) SFR Reset value: 0000h

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|---------|--------|----|---------|---------|---|------|-------|-------|------|---|---|---|---|
| CSWEN1 | CSREN1 | RDYPOL1 | RDYEN1 | - | BUSACT1 | ALECTL1 | - | BTYP | MTTC1 | RWDC1 | MCTC | | | | |
| RW | RW | RW | RW | | RW | RW | | RW | RW | RW | RW | | | | |

BUSCON2 (FF16h / 8Bh) SFR Reset value: 0000h

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|---------|--------|----|---------|---------|---|------|-------|-------|------|---|---|---|---|
| CSWEN2 | CSREN2 | RDYPOL2 | RDYEN2 | - | BUSACT2 | ALECTL2 | - | BTYP | MTTC2 | RWDC2 | MCTC | | | | |
| RW | RW | | RW | | RW | RW | | RW | RW | RW | RW | | | | |

BUSCON3 (FF18h / 8Ch) SFR Reset value: 0000h

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------|--------|---------|--------|----|---------|---------|---|------|-------|-------|------|---|---|---|---|
| CSWEN3 | CSREN3 | RDYPOL3 | RDYEN3 | - | BUSACT3 | ALECTL3 | - | BTYP | MTTC3 | RWDC3 | MCTC | | | | |
| RW | RW | | RW | | RW | RW | | RW | RW | RW | RW | | | | |

| BUSCON4 (FF1Ah / 8Dh) | | | | | | | | | | SFR | | | | Reset value: 0000h | | | |
|-----------------------|--------|---------|--------|----|---------|---------|---|------|-------|-------|---|---|---|--------------------|---|--|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| CSWEN4 | CSREN4 | RDYPOL4 | RDYEN4 | - | BUSACT4 | ALECTL4 | - | BTYP | MTTC4 | RWDC4 | | | | | | | |
| RW | RW | | RW | | RW | RW | | RW | RW | RW | | | | | | | |

Table 78. BUSCON4 description

| Bit | Function |
|---------|--|
| MCTC | Memory cycle time control (number of memory cycle time wait-states) '0000': 15 wait-states (Number of wait-states = 15 - [MCTC]). ... '1111': No wait-states. |
| RWDCx | Read/Write delay control for BUSCONx '0': With read/write delay, the CPU inserts 1 TCL after falling edge of ALE. '1': No read/write delay, RW is activated after falling edge of ALE. |
| MTTCx | Memory tristate time control '0': 1 wait-state. '1': No wait-state. |
| BTYP | External bus configuration '00': 8-bit Demultiplexed Bus '01': 8-bit Multiplexed Bus '10': 16-bit Demultiplexed Bus '11': 16-bit Multiplexed Bus Note: For BUSCON0 BTYP is defined via PORT0 during reset. |
| ALECTLx | ALE lengthening control '0': Normal ALE signal. '1': Lengthened ALE signal. |
| BUSACTx | Bus active control '0': External bus disabled. '1': External bus enabled (within the respective address window, see ADDRSEL). |
| RDYENx | Ready input enable '0': External bus cycle is controlled by bit field MCTC only. '1': External bus cycle is controlled by the $\overline{\text{READY}}$ input signal. |
| RDYPOLx | Ready active level control '0': Active level on the READY pin is low, bus cycle terminates with a '0' on READY pin. '1': Active level on the READY pin is high, bus cycle terminates with a '1' on READY pin. |
| CSRENx | Read chip select enable '0': The $\overline{\text{CS}}$ signal is independent of the read command ($\overline{\text{RD}}$). '1': The $\overline{\text{CS}}$ signal is generated for the duration of the read command. |
| CSWENx | Write chip select enable '0': The $\overline{\text{CS}}$ signal is independent of the write command ($\overline{\text{WR}}$, $\overline{\text{WRL}}$, $\overline{\text{WRH}}$). '1': The $\overline{\text{CS}}$ signal is generated for the duration of the write command. |

- Note: 1 *BTYP (bit 6 and 7) is set according to the configuration of the bit I6 and I7 of PORT0 latched at the end of the reset sequence.*
- 2 *BUSCON0 is initialized with 0000h, if \overline{EA} pin is high during reset. If \overline{EA} pin is low during reset, bit BUSACT0 and ALECTRL0 are set ('1') and bit field BTYP is loaded with the bus configuration selected via PORT0.*

| RP0H (F108h / 84h) | | | | | | | | ESFR | | | | Reset value: --XXh | | | |
|--------------------|----|----|----|----|----|---|---|--------|---|--------|---|--------------------|---|-----|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | | | | | | | | CLKSEL | | SALSEL | | CSSEL | | WRC | |
| | | | | | | | | R | | R | | R | | R | |

Table 79. RPOH description⁽¹⁾

| Bit | Function |
|---------------------------|---|
| WRC ⁽²⁾ | Write configuration control '0': Pin \overline{WR} acts as \overline{WRL} , pin \overline{BHE} acts as \overline{WRH} '1': Pins \overline{WR} and \overline{BHE} retain their normal function |
| CSSEL ⁽²⁾ | Chip select line selection (number of active \overline{CS} outputs) 0 0: 3 \overline{CS} lines: $\overline{CS}2... \overline{CS}0$ 0 1: 2 \overline{CS} lines: $\overline{CS}1... \overline{CS}0$ 1 0: No \overline{CS} line at all 1 1: 5 \overline{CS} lines: $\overline{CS}4... \overline{CS}0$ (Default without pull-downs) |
| SALSEL ⁽²⁾ | Segment address line selection (number of active segment address outputs) '00': 4-bit segment address: A19...A16 '01': No segment address lines at all '10': 8-bit segment address: A23...A16 '11': 2-bit segment address: A17...A16 (Default without pull-downs) |
| CLKSEL ^{(2) (3)} | System clock selection '000': $f_{CPU} = 16 \times f_{OSC}$ '001': $f_{CPU} = 0.5 \times f_{OSC}$ '010': $f_{CPU} = 10 \times f_{OSC}$ '011': $f_{CPU} = f_{OSC}$ '100': $f_{CPU} = 5 \times f_{OSC}$ '101': $f_{CPU} = 8 \times f_{OSC}$ '110': $f_{CPU} = 3 \times f_{OSC}$ '111': $f_{CPU} = 4 \times f_{OSC}$ |

1. RP0H is a read-only register.
2. These bits are set according to Port 0 configuration during any reset sequence.
3. RP0H.7 to RP0H.5 bits are loaded only during a long hardware reset. As pull-up resistors are active on each Port P0H pins during reset, RP0H default value is "FFh".

| EXICON (F1C0h / E0h) | | | | | | | | ESFR | | | | Reset value: 0000h | | | |
|----------------------|--------|--------|--------|--------|--------|--------|--------|------|---|---|---|--------------------|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EXI7ES | EXI6ES | EXI5ES | EXI4ES | EXI3ES | EXI2ES | EXI1ES | EXI0ES | | | | | | | | |
| RW | RW | RW | RW | RW | RW | RW | RW | | | | | | | | |

Table 80. EXIxES bit description

| Bit | Function |
|---------------------|---|
| EXIxES (x=7...0) | 00 = Fast external interrupts disabled: Standard mode. EXxIN pin not taken in account for entering/exiting Power Down mode. 01 = Interrupt on positive edge (rising). Enter Power Down mode if EXiIN = '0', exit if EXxIN = '1' (referred as "high" active level) 10 = Interrupt on negative edge (falling). Enter Power Down mode if EXiIN = '1', exit if EXxIN = '0' (referred as "low" active level) 11 = Interrupt on any edge (rising or falling). Always enter Power Down mode, exit if EXxIN level changed. |

| EXISEL (F1DAh / EDh) | | | | | | ESFR | | | | | | Reset value: 0000h | | | |
|----------------------|--------|--------|--------|--------|--------|--------|--------|---|---|---|---|--------------------|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| EXI7SS | EXI6SS | EXI5SS | EXI4SS | EXI3SS | EXI2SS | EXI1SS | EXI0SS | | | | | | | | |
| RW | RW | RW | RW | RW | RW | RW | RW | | | | | | | | |

Table 81. EXISEL

| Bit | Function |
|--------|---|
| EXIxSS | External Interrupt x Source Selection (x = 7...0) 00 = Input from associated Port 2 pin. 01 = Input from "alternate source". 10 = Input from Port 2 pin ORed with "alternate source". 11 = Input from Port 2 pin ANDed with "alternate source". |

Table 82. EXIxSS and port 2 pin configurations

| EXIxSS | Port 2 pin | Alternate source |
|--------|------------|------------------|
| 0 | P2.8 | CAN1_RxD |
| 1 | P2.9 | CAN2_RxD / SCL |
| 2 | P2.10 | RTCSI (Second) |
| 3 | P2.11 | RTCAI (Alarm) |
| 4...7 | P2.12...15 | Not used (zero) |

| XP3IC (F19Eh / CFh) | | | | | | ESFR | | | | | | Reset value: --00h | | | |
|---------------------|----|----|----|----|----|------|---|-------|-------|---------|---|--------------------|------|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | - | - | - | - | - | XP3IR | XP3IE | XP3ILVL | | | GLVL | | |
| | | | | | | | | RW | RW | RW | | | RW | | |

Note: 1. XP3IC register has the same bit field as xxIC interrupt registers

| xxIC (yyyyh / zzh) | | | | | | SFR area | | | | | | Reset value: --00h | | | |
|--------------------|----|----|----|----|----|----------|---|------|------|------|---|--------------------|------|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| - | - | - | - | - | - | - | - | xxIR | xxIE | ILVL | | | GLVL | | |
| | | | | | | | | RW | RW | RW | | | RW | | |

Table 83. SFR area description

| Bit | Function |
|------|---|
| GLVL | Group level Defines the internal order for simultaneous requests of the same priority. '3': Highest group priority '0': Lowest group priority |
| ILVL | Interrupt priority level Defines the priority level for the arbitration of requests. 'Fh': Highest priority level '0h': Lowest priority level |
| xxIE | Interrupt enable control bit (individually enables/disables a specific source) '0': Interrupt request is disabled '1': Interrupt request is enabled |
| xxIR | Interrupt request flag '0': No request pending '1': This source has raised an interrupt request |

| XPERCON (F024h / 12h) | | | | | ESFR | | | | | | | | Reset value:- 005h | | | |
|-----------------------|----|----|----|----|-------------|------------|------------|------------|------------|--------------|------------|-------------|--------------------|------------|------------|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| - | - | - | - | - | XMISC EN | XI2C EN | XSSC EN | XASC EN | XPWM EN | XFLAS HEN | XRTC EN | XRAM2 EN | XRAM1 EN | CAN2 EN | CAN1 EN | |
| - | - | - | - | - | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | |

Table 84. ESFR description

| Bit | Function |
|---------|--|
| CAN1EN | CAN1 enable bit '0': Accesses to the on-chip CAN1 XPeripheral and its functions are disabled (P4.5 and P4.6 pins can be used as general purpose I/Os, but address range 00'EC00h-00'EFFFh is directed to external memory only if CAN2EN, XRTCEN, XASCEN, XSSCEN, XI2CEN, XPWMEN and XMISCEN are '0' also). '1': The on-chip CAN1 XPeripheral is enabled and can be accessed. |
| CAN2EN | CAN2 enable bit '0': Accesses to the on-chip CAN2 XPeripheral and its functions are disabled (P4.4 and P4.7 pins can be used as general purpose I/Os, but address range 00'EC00h-00'EFFFh is directed to external memory only if CAN1EN, XRTCEN, XASCEN, XSSCEN, XI2CEN, XPWMEN and XMISCEN are '0' also). '1': The on-chip CAN2 XPeripheral is enabled and can be accessed. |
| XRAM1EN | XRAM1 enable bit '0': Accesses to the on-chip 2 Kbyte XRAM are disabled. Address range 00'E000h-00'E7FFh is directed to external memory. '1': The on-chip 2 Kbyte XRAM is enabled and can be accessed. |
| XRAM2EN | XRAM2 enable bit '0': Accesses to the on-chip 64 Kbyte XRAM are disabled, external access performed. Address range 0F'0000h-0F'FFFFh is directed to external memory only if XFLASHEN is '0' also. '1': The on-chip 64 Kbyte XRAM is enabled and can be accessed. |

Table 84. ESFR description (continued)

| Bit | Function |
|----------|---|
| XRTCEN | <p>RTC enable</p> <p>'0': Accesses to the on-chip RTC module are disabled, external access performed. Address range 00'ED00h-00'EDFF is directed to external memory only if CAN1EN, CAN2EN, XASCEN, XSSCEN, XI2CEN, XPWMEN and XMISCEN are '0' also.</p> <p>'1': The on-chip RTC module is enabled and can be accessed.</p> |
| XPWMEN | <p>XPWM enable</p> <p>'0': Accesses to the on-chip XPWM module are disabled, external access performed. Address range 00'EC00h-00'ECFF is directed to external memory only if CAN1EN, CAN2EN, XASCEN, XSSCEN, XI2CEN, XRTCEN and XMISCEN are '0' also.</p> <p>'1': The on-chip XPWM module is enabled and can be accessed.</p> |
| XFLASHEN | <p>XFlash enable bit</p> <p>'0': Accesses to the on-chip XFlash and Flash registers are disabled, external access performed. Address range 09'0000h-0E'FFFFh is directed to external memory only if XRAM2EN is '0' also.</p> <p>'1': The on-chip XFlash is enabled and can be accessed.</p> |
| XASCEN | <p>XASC enable bit</p> <p>'0': Accesses to the on-chip XASC are disabled, external access performed. Address range 00'E900h-00'E9FFh is directed to external memory only if CAN1EN, CAN2EN, XRTCEN, XASCEN, XI2CEN, XPWMEN and XMISCEN are '0' also.</p> <p>'1': The on-chip XASC is enabled and can be accessed.</p> |
| XSSCEN | <p>XSSC enable bit</p> <p>'0': Accesses to the on-chip XSSC are disabled, external access performed. Address range 00'E800h-00'E8FFh is directed to external memory only if CAN1EN, CAN2EN, XRTCEN, XASCEN, XI2CEN, XPWMEN and XMISCEN are '0' also.</p> <p>'1': The on-chip XSSC is enabled and can be accessed.</p> |
| XI2CEN | <p>I²C enable bit</p> <p>'0': Accesses to the on-chip I²C are disabled, external access performed. Address range 00'EA00h-00'EAFFh is directed to external memory only if CAN1EN, CAN2EN, XRTCEN, XASCEN, XSSCEN, XPWMEN and XMISCEN are '0' also.</p> <p>'1': The on-chip I²C is enabled and can be accessed.</p> |
| XMISCEN | <p>XBUS additional features enable bit</p> <p>'0': Accesses to the Additional Miscellaneous Features is disabled. Address range 00'EB00h-00'EBFFh is directed to external memory only if CAN1EN, CAN2EN, XRTCEN, XASCEN, XSSCEN, XPWMEN and XI2CEN are '0' also.</p> <p>'1': The Additional Features are enabled and can be accessed.</p> |

When CAN1, CAN2, RTC, XASC, XSSC, I²C, XPWM and the XBUS Additional Features are all disabled via XPERCON setting, then any access in the address range 00'E800h - 00'EBFFh is directed to external memory interface, using the BUSCONx register corresponding to the address matching ADDRSELx register. All pins used for X-Peripherals can be used as General Purpose I/O whenever the related module is not enabled.

The default XPER selection after Reset is such that CAN1 is enabled, CAN2 is disabled, XRAM1 (2 Kbyte XRAM) is enabled and XRAM2 (64 Kbyte XRAM) is disabled; all the other X-Peripherals are disabled after Reset.

Register XPERCON cannot be changed after the global enabling of X-Peripherals, that is, after setting of bit XPEN in SYSCON register.

In Emulation mode, all the X-Peripherals are enabled (XPERCON bits are all set). The bondout chip determines whether or not to redirect an access to external memory or to XBUS.

Reserved bits of XPERCON register are always written to '0'.

[Table 85](#) below summarizes the Segment 8 mapping that depends upon the \overline{EA} pin status during reset as well as the SYSCON (bit XPEN) and XPERCON (bits XRAM2EN and XFLASHEN) registers user programmed values.

Table 85. Segment 8 address range mapping

| \overline{EA} | XPEN | XRAM2EN | XFLASHEN | Segment 8 |
|-----------------|------|---------|----------|-----------------|
| 0 | 0 | x | x | External memory |
| 0 | 1 | 0 | 0 | External memory |
| 0 | 1 | 1 | x | Reserved |
| 0 | 1 | x | 1 | Reserved |
| 1 | x | x | x | IFlash (B1F1) |

Note: The symbol "x" in the table above stands for "do not care".

22.10.1 XPERCON and XPEREMU registers

As already mentioned, the XPERCON register must be programmed to enable the single XBUS modules separately. The XPERCON is a read/write ESFR register; the XPEREMU register is a write-only register mapped on XBUS memory space (address EB7Eh).

Once the XPEN bit of SYSCON register is set and at least one of the X-peripherals (except memories) is activated, the register XPEREMU must be written with the same content of XPERCON: This is mandatory in order to allow a correct emulation of the new set of features introduced on XBUS for the new ST10 generation. The following instructions must be added inside the initialization routine:

```
if (SYSCON.XPEN && (XPERCON & 0x07D3))
then { XPEREMU = XPERCON }
```

Of course, XPEREMU must be programmed after XPERCON and after SYSCON; in this way the final configuration for X-Peripherals is stored in XPEREMU and used for the emulation hardware setup.

| XPEREMU (EB7Eh) | | | | | XBUS | | | | | | | | Reset value xxxhx: | | | |
|-----------------|----|----|----|----|-------------|------------|------------|------------|------------|--------------|------------|-------------|--------------------|------------|------------|--|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| - | - | - | - | - | XMISC EN | XI2C EN | XSSC EN | XASC EN | XPWM EN | XFLAS HEN | XRTC EN | XRAM2 EN | XRAM1 EN | CAN2 EN | CAN1 EN | |
| - | - | - | - | - | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | RW | |

Note: The bit meaning is exactly the same as in XPERCON.

22.11 Emulation dedicated registers

Four additional registers are implemented for emulation purposes only. Similarly to XPEREMU, they are write-only registers.

XEMU0 (EB76h) XBUS Reset value: xxxh

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| XEMU0(15:0) | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | |

XEMU1 (EB78h) XBUS Reset value: xxxh

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| XEMU1(15:0) | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | |

XEMU2 (EB7Ah) XBUS Reset value: xxxh:

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| XEMU2(15:0) | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | |

XEMU3 (EB7Ch) XBUS Reset value: xxxh

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| XEMU3(15:0) | | | | | | | | | | | | | | | |
| W | | | | | | | | | | | | | | | |

23 Electrical characteristics

23.1 Absolute maximum ratings

Table 86. Absolute maximum ratings

| Symbol | Parameter | Value | Unit |
|-------------------|--|--------------------------------|------|
| V _{DD} | Voltage on V _{DD} pins with respect to ground (V _{SS}) | - 0.3 to +6.5 | V |
| V _{STBY} | Voltage on V _{STBY} pin with respect to ground (V _{SS}) | | |
| V _{AREF} | Voltage on V _{AREF} pin with respect to ground (V _{SS}) | - 0.3 to V _{DD} + 0.3 | |
| V _{AGND} | Voltage on V _{AGND} pin with respect to ground (V _{SS}) | V _{SS} | |
| V _{IO} | Voltage on any pin with respect to ground (V _{SS}) | - 0.5 to V _{DD} + 0.5 | |
| I _{OV} | Input current on any pin during overload condition | ± 10 | mA |
| I _{TOV} | Absolute sum of all input currents during overload condition | 75 | |
| T _{ST} | Storage temperature | - 65 to +150 | °C |
| ESD | ESD susceptibility (human body model) | 2000 | V |

Note: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. During overload conditions ($V_{IN} > V_{DD}$ or $V_{IN} < V_{SS}$) the voltage on pins with respect to ground (V_{SS}) must not exceed the values defined by the Absolute Maximum Ratings.

During Power-on and Power-off transients (including Stand-by entering/exiting phases), the relationships between voltages applied to the device and the main V_{DD} must always be respected. In particular, power-on and power-off of V_{AREF} must be coherent with the V_{DD} transient, in order to avoid undesired current injection through the on-chip protection diodes.

23.2 Recommended operating conditions

Table 87. Recommended operating conditions

| Symbol | Parameter | Min. | Max. | Unit |
|-------------------|---|------|-----------------|------|
| V _{DD} | Operating supply voltage | 4.5 | 5.5 | V |
| V _{STBY} | Operating stand-by supply voltage ⁽¹⁾ | | | |
| V _{AREF} | Operating analog reference voltage ⁽²⁾ | 0 | V _{DD} | |
| T _A | Ambient temperature under bias | -40 | +125 | °C |
| T _J | Junction temperature under bias | | +150 | |

1. The value of the V_{STBY} voltage is specified in the range 4.5 - 5.5 volts. Nevertheless, it is acceptable to exceed the upper limit (up to 6.0 volts) for a maximum of 100 hours over the global 300000 hours, representing the lifetime of the device (about 30 years). On the other hand, it is possible to exceed the lower limit (down to 4.0 volts) whenever RTC and 32 kHz on-chip oscillator amplifier are turned off (only Stand-by RAM powered through V_{STBY} pin in Stand-by mode). When V_{STBY} voltage is lower than main V_{DD}, the input section of V_{STBY}/EA pin can generate a spurious static consumption on V_{DD} power supply (in the range of tenth of μ A).

2. For details on operating conditions concerning the usage of A/D converter, refer to [Section 23.7](#).

23.3 Power considerations

The average chip-junction temperature, T_J , in degrees Celsius, is calculated using the following equation:

$$T_J = T_A + (P_D \times \Theta_{JA}) \quad (1)$$

Where:

T_A is the Ambient Temperature in °C,

Θ_{JA} is the Package Junction-to-Ambient Thermal Resistance, in °C/W,

P_D is the sum of P_{INT} and $P_{I/O}$ ($P_D = P_{INT} + P_{I/O}$),

P_{INT} is the product of I_{DD} and V_{DD} , expressed in Watt. This is the Chip Internal Power,

$P_{I/O}$ represents the Power Dissipation on Input and Output Pins; user determined.

Most often in applications, $P_{I/O} < P_{INT}$, which may be ignored. On the other hand, $P_{I/O}$ may be significant if the device is configured to continuously drive external modules and/or memories.

An approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected) is given by:

$$P_D = K / (T_J + 273^\circ\text{C}) \quad (2)$$

Therefore (solving equations 1 and 2):

$$K = P_D \times (T_A + 273^\circ\text{C}) + \Theta_{JA} \times P_D^2 \quad (3)$$

Where:

K is a constant for the particular part, which may be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A . Using this value of K , the values of P_D and T_J are obtained by solving equations (1) and (2) iteratively for any value of T_A .

Table 88. Thermal characteristics

| Symbol | Description | Value (typical) | Unit |
|---------------|--|-----------------|------|
| Θ_{JA} | Thermal resistance junction-ambient | | |
| | PQFP 144 - 28 x 28 x 3.4 mm / 0.65 mm pitch | 30 | °C/W |
| | LQFP 144 - 20 x 20 mm / 0.5 mm pitch | 40 | |
| | LQFP 144 - 20 x 20 mm / 0.5 mm pitch on four layer FR4 board (2 layers signals / 2 layers power) | 35 | |

Based on thermal characteristics of the package and with reference to the power consumption figures provided in the next tables and diagrams, the following product classification can be proposed. In any case, the exact power consumption of the device inside the application must be computed according to different working conditions, thermal profiles, real thermal resistance of the system (including printed circuit board or other substrata) and I/O activity.

Table 89. Package characteristics

| Package | Operating temperature | CPU frequency range |
|----------|-----------------------|---------------------|
| Die | - 40 / +125°C | 1 – 64 MHz |
| PQFP 144 | | 1 – 40 MHz |
| LQFP 144 | | |
| LQFP 144 | -40/+105°C | 1 – 48 MHz |

23.4 Parameter interpretation

The parameters listed in the following tables represent the characteristics of the ST10F276 and its demands on the system.

Where the ST10F276 logic provides signals with their respective timing characteristics, the symbol “CC” (Controller Characteristics) is included in the “Symbol” column. Where the external system must provide signals with their respective timing characteristics to the ST10F276, the symbol “SR” (System Requirement) is included in the “Symbol” column.

23.5 DC characteristics

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$

Table 90. DC characteristics

| Symbol | Parameter | Test Condition | Limit values | | Unit |
|--------------|--|----------------------|--------------|----------------|------|
| | | | Min. | Max. | |
| V_{IL} SR | Input low voltage (TTL mode) (except \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD, XTAL1, READY) | – | – 0.3 | 0.8 | V |
| V_{ILS} SR | Input low voltage (CMOS mode) (except \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD, XTAL1, READY) | – | – 0.3 | $0.3 V_{DD}$ | |
| V_{IL1} SR | Input low voltage \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD | – | – 0.3 | $0.3 V_{DD}$ | |
| V_{IL2} SR | Input low voltage XTAL1 (CMOS only) | Direct drive mode | – 0.3 | $0.3 V_{DD}$ | |
| V_{IL3} SR | Input low voltage READY (TTL only) | – | – 0.3 | 0.8 | |
| V_{IH} SR | Input high voltage (TTL mode) (except \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD, XTAL1) | – | 2.0 | $V_{DD} + 0.3$ | |
| V_{IHS} SR | Input high voltage (CMOS mode) (except \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD, XTAL1) | – | $0.7 V_{DD}$ | $V_{DD} + 0.3$ | |
| V_{IH1} SR | Input high voltage \overline{RSTIN} , \overline{EA} , \overline{NMI} , RPD | – | $0.7 V_{DD}$ | $V_{DD} + 0.3$ | |

Table 90. DC characteristics (continued)

| Symbol | Parameter | Test Condition | Limit values | | Unit |
|-----------------------|--|---|---|---|---------|
| | | | Min. | Max. | |
| V _{IH2} SR | Input high voltage XTAL1 (CMOS only) | Direct Drive mode | 0.7 V _{DD} | V _{DD} + 0.3 | V |
| V _{IH3} SR | Input high voltage READY (TTL only) | – | 2.0 | V _{DD} + 0.3 | |
| VHYS CC | Input hysteresis (TTL mode) (except $\overline{\text{RSTIN}}$, $\overline{\text{EA}}$, $\overline{\text{NMI}}$, XTAL1, RPD) | 3) | 400 | 700 | mV |
| VHYSSCC | Input Hysteresis (CMOS mode) (except $\overline{\text{RSTIN}}$, $\overline{\text{EA}}$, $\overline{\text{NMI}}$, XTAL1, RPD) | 3) | 750 | 1400 | |
| VHYS1CC | Input hysteresis $\overline{\text{RSTIN}}$, $\overline{\text{EA}}$, $\overline{\text{NMI}}$ | 3) | 750 | 1400 | |
| VHYS2CC | Input hysteresis XTAL1 | 3) | 0 | 50 | |
| VHYS3CC | Input hysteresis READY (TTL only) | 3) | 400 | 700 | |
| VHYS4CC | Input hysteresis RPD | 3) | 500 | 1500 | |
| V _{OL} CC | Output low voltage (P6[7:0], ALE, RD, $\overline{\text{WR/WRL}}$, $\overline{\text{BHE/WRH}}$, CLKOUT, $\overline{\text{RSTIN}}$, RSTOUT) | I _{OL} = 8 mA I _{OL} = 1 mA | – | 0.4 0.05 | V |
| V _{OL1} CC | Output low voltage (P0[15:0], P1[15:0], P2[15:0], P3[15,13:0], P4[7:0], P7[7:0], P8[7:0]) | I _{OL1} = 4 mA I _{OL1} = 0.5 mA | – | 0.4 0.05 | |
| V _{OL2} CC | Output low voltage RPD | I _{OL2} = 85 μ A I _{OL2} = 80 μ A I _{OL2} = 60 μ A | – | V _{DD} 0.5 V _{DD} 0.3 V _{DD} | |
| V _{OH} CC | Output high voltage (P6[7:0], ALE, RD, $\overline{\text{WR/WRL}}$, $\overline{\text{BHE/WRH}}$, CLKOUT, $\overline{\text{RSTOUT}}$) | I _{OH} = – 8 mA I _{OH} = – 1 mA | V _{DD} – 0.8 V _{DD} – 0.08 | – | |
| V _{OH1} CC | Output high voltage ⁽¹⁾ (P0[15:0], P1[15:0], P2[15:0], P3[15,13:0], P4[7:0], P7[7:0], P8[7:0]) | I _{OH1} = – 4 mA I _{OH1} = – 0.5 mA | V _{DD} – 0.8 V _{DD} – 0.08 | – | |
| V _{OH2} CC | Output high voltage RPD | I _{OH2} = – 2 mA I _{OH2} = – 750 μ A I _{OH2} = – 150 μ A | 0 0.3 V _{DD} 0.5 V _{DD} | – | |
| I _{OZ1} CC | Input leakage current (P5[15:0]) ⁽²⁾ | – | – | ±0.2 | μ A |
| I _{OZ2} CC | Input leakage current (all except P5[15:0], P2.0, RPD) | – | – | ±0.5 | |
| I _{OZ3} CC | Input leakage current (P2.0) ⁽³⁾ | – | – | +1.0 –0.5 | |
| I _{OZ4} CC | Input leakage current (RPD) | – | – | ±3.0 | |
| I _{OV1} SR | Overload current (all except P2.0) | (4) (5) | – | ±5 | mA |

Table 90. DC characteristics (continued)

| Symbol | Parameter | Test Condition | Limit values | | Unit |
|-----------------------|--|---|--------------|--------------------|------------|
| | | | Min. | Max. | |
| $ I_{OV2} $ SR | Overload current (P2.0) ⁽³⁾ | (4)(5) | – | +5 –1 | mA |
| R_{RST} CC | \overline{RSTIN} pull-up resistor | 100 k Ω nominal | 50 | 250 | k Ω |
| I_{RWH} | Read/Write inactive current ^{(6) (7)} | $V_{OUT} = 2.4$ V | – | –40 | μ A |
| I_{RWL} | Read/Write active current ⁽⁶⁾⁽⁸⁾ | $V_{OUT} = 0.4$ V | –500 | – | |
| I_{ALEL} | ALE inactive current ^{(6) (7)} | $V_{OUT} = 0.4$ V | 20 | – | |
| I_{ALEH} | ALE active current ^{(6) (8)} | $V_{OUT} = 2.4$ V | – | 300 | |
| I_{P6H} | Port 6 inactive current (P6[4:0]) ⁽⁶⁾⁽⁷⁾ | $V_{OUT} = 2.4$ V | – | –40 | |
| I_{P6L} | Port 6 active current (P6[4:0]) ^{(6) (8)} | $V_{OUT} = 0.4$ V | –500 | – | |
| $I_{P0H}^{(6)}$ | PORT0 configuration current ⁽⁶⁾ | $V_{IN} = 2.0$ V | – | –10 | |
| $I_{P0L}^{(7)}$ | | $V_{IN} = 0.8$ V | –100 | – | |
| C_{IO} CC | Pin capacitance (digital inputs / outputs) | (4)(6) | – | 10 | pF |
| I_{CC1} | Run mode power supply current ⁽⁹⁾ (execution from internal RAM) | – | – | $20 + 2 f_{CPU}$ | mA |
| I_{CC2} | Run mode power supply current ⁽⁴⁾⁽¹⁰⁾ (execution from internal Flash) | – | – | $20 + 1.8 f_{CPU}$ | mA |
| I_{ID} | Idle mode supply current ⁽¹¹⁾ | – | – | $20 + 0.6 f_{CPU}$ | mA |
| I_{PD1} | Power Down supply current ⁽¹²⁾ (RTC off, oscillators off, main voltage regulator off) | $T_A = 25^\circ\text{C}$ | – | 1 | mA |
| I_{PD2} | Power Down supply current ⁽¹²⁾ (RTC on, main oscillator on, main voltage regulator off) | $T_A = 25^\circ\text{C}$ | – | 8 | mA |
| I_{PD3} | Power down supply current ⁽¹²⁾ (RTC on, 32 kHz oscillator on, main voltage regulator off) | $T_A = 25^\circ\text{C}$ | – | 1.1 | mA |
| I_{SB1} | Stand-by supply current ⁽¹²⁾ (RTC off, Oscillators off, VDD off, VSTBY on) | $V_{STBY} = 5.5$ V $T_A = T_J = 25^\circ\text{C}$ | – | 250 | μ A |
| | | $V_{STBY} = 5.5$ V $T_A = T_J = 125^\circ\text{C}$ | – | 500 | μ A |
| I_{SB2} | Stand-by supply current ⁽¹²⁾ (RTC on, 32 kHz Oscillator on, main VDD off, VSTBY on) | $V_{STBY} = 5.5$ V $T_A = 25^\circ\text{C}$ | – | 250 | μ A |
| | | $V_{STBY} = 5.5$ V $T_A = 125^\circ\text{C}$ | – | 500 | μ A |
| I_{SB3} | Stand-by supply current ^{(4) (12)} (VDD transient condition) | – | – | 2.5 | mA |

1. This specification is not valid for outputs which are switched to open drain mode. In this case the respective output floats and the voltage is imposed by the external circuitry.
2. Port 5 leakage values are granted for not selected A/D converter channel. One channels is always selected (by default, after reset, P5.0 is selected). For the selected channel the leakage value is similar to that of other port pins.
3. The leakage of P2.0 is higher than other pins due to the additional logic (pass gates active only in specific test modes) implemented on input path. Pay attention to not stress P2.0 input pin with negative overload beyond the specified limits: Failures in Flash reading may occur (sense amplifier perturbation). Refer to next Figure 44 for a scheme of the input circuitry.
4. Not 100% tested, guaranteed by design characterization.
5. Overload conditions occur if the standard operating conditions are exceeded, that is, the voltage on any pin exceeds the specified range (that is, $V_{OV} > V_{DD} + 0.3V$ or $V_{OV} < -0.3V$). The absolute sum of input overload currents on all port pins may not exceed 50mA. The supply voltage must remain within the specified limits.
6. This specification is only valid during Reset, or during Hold- or Adapt-mode. Port 6 pins are only affected if they are used for CS output and the open drain function is not enabled.
7. The maximum current may be drawn while the respective signal line remains inactive.
8. The minimum current must be drawn in order to drive the respective signal line active.
9. The power supply current is a function of the operating frequency (f_{CPU} is expressed in MHz). This dependency is illustrated in the Figure 45 below. This parameter is tested at V_{DDmax} and at maximum CPU clock frequency with all outputs disconnected and all inputs at V_{IL} or V_{IH} , RSTIN pin at V_{IH1min} : **This implies I/O current is not considered.** The device is doing the following:
 - Fetching code from IRAM and XRAM1, accessing in read and write to both XRAM modules
 - Watchdog Timer is enabled and regularly serviced
 - RTC is running with main oscillator clock as reference, generating a tick interrupts every 192 clock cycles
 - Four channels of XPWM are running (waves period: 2, 2.5, 3 and 4 CPU clock cycles): No output toggling
 - Five General Purpose Timers are running in timer mode with prescaler equal to 8 (T2, T3, T4, T5, T6)
 - ADC is in Auto Scan Continuous Conversion mode on all 16 channels of Port5
 - All interrupts generated by XPWM, RTC, Timers and ADC are not serviced
10. The power supply current is a function of the operating frequency (f_{CPU} is expressed in MHz). This dependency is illustrated in the Figure 45 below. This parameter is tested at V_{DDmax} and at maximum CPU clock frequency with all outputs disconnected and all inputs at V_{IL} or V_{IH} , RSTIN pin at V_{IH1min} : **This implies I/O current is not considered.** The device is doing the following:
 - Fetching code from all sectors of both IFlash and XFlash, accessing in read (few fetches) and write to XRAM
 - Watchdog Timer is enabled and regularly serviced
 - RTC is running with main oscillator clock as reference, generating a tick interrupts every 192 clock cycles
 - Four channels of XPWM are running (waves period: 2, 2.5, 3 and 4 CPU clock cycles): No output toggling
 - Five General Purpose Timers are running in timer mode with prescaler equal to 8 (T2, T3, T4, T5, T6)
 - ADC is in Auto Scan Continuous Conversion mode on all 16 channels of Port5
 - All interrupts generated by XPWM, RTC, Timers and ADC are not serviced
11. The Idle mode supply current is a function of the operating frequency (f_{CPU} is expressed in MHz). This dependency is illustrated in the Figure 44 below. These parameters are tested and at maximum CPU clock with all outputs disconnected and all inputs at V_{IL} or V_{IH} , RSTIN pin at V_{IH1min} .
12. This parameter is tested including leakage currents. All inputs (including pins configured as inputs) at 0 to 0.1V or at $V_{DD} - 0.1V$ to V_{DD} , $V_{AREF} = 0V$, all outputs (including pins configured as outputs) disconnected. Furthermore, the Main Voltage Regulator is assumed off: In case it is not, additional 1mA shall be assumed.

Figure 44. Port2 test mode structure

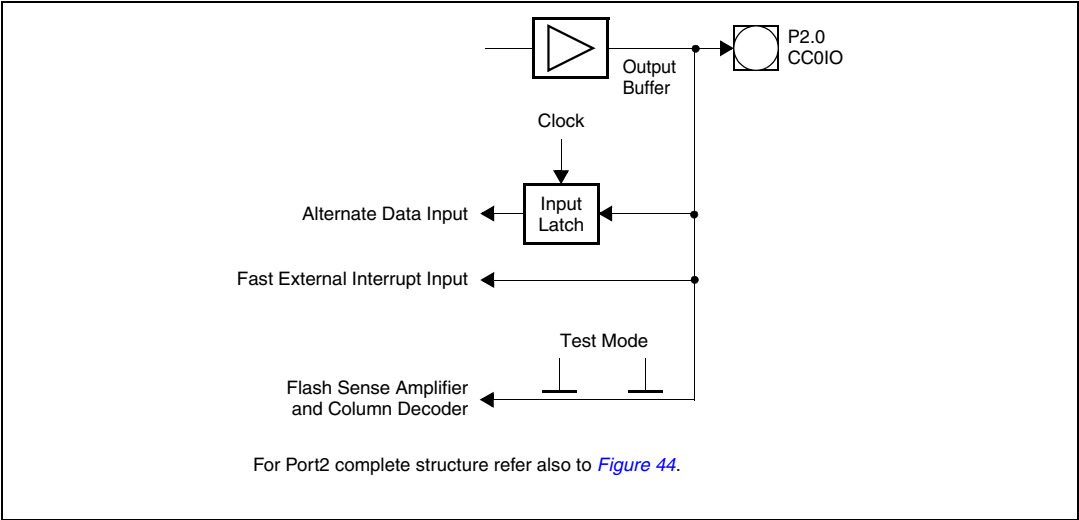
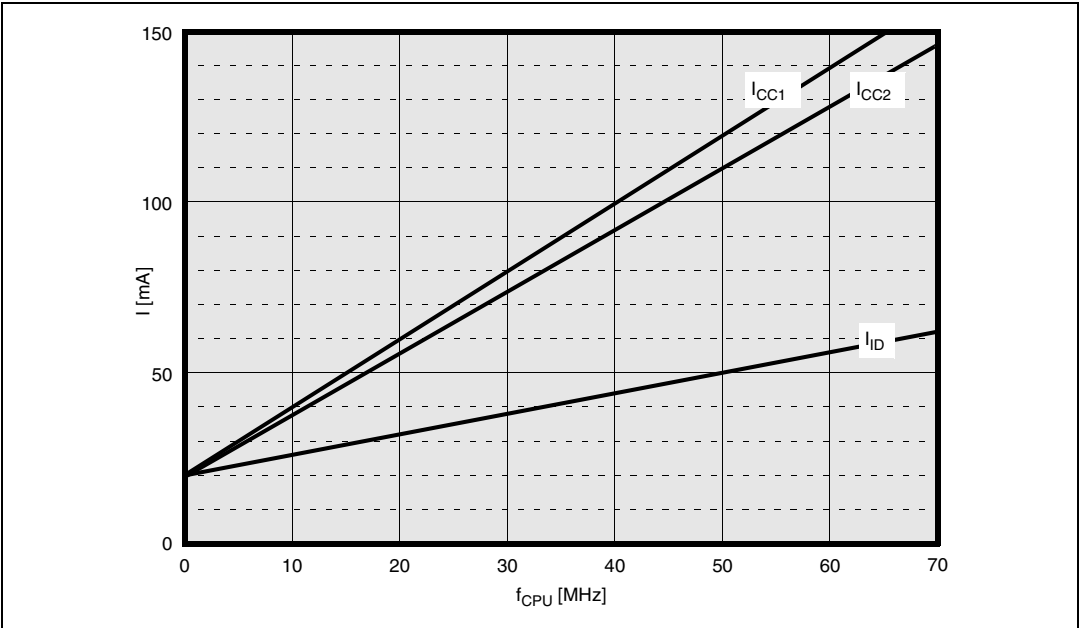


Figure 45. Supply current versus the operating frequency (RUN and IDLE modes)



23.6 Flash characteristics

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$

Table 91. Flash characteristics

| Parameter | Typical $T_A = 25^\circ C$ | Maximum $T_A = 125^\circ C$ | | Unit | Notes |
|--|-------------------------------|--------------------------------|--------------|---------|------------------------------------|
| | 0 cycles ⁽¹⁾ | 0 cycles ⁽¹⁾ | 100k cycles | | |
| Word program (32-bit) ⁽²⁾ | 35 | 80 | 290 | μs | — |
| Double word program (64-bit) ⁽²⁾ | 60 | 150 | 570 | μs | — |
| Bank 0 program (384K) (double word program) | 2.9 | 7.4 | 28.0 | s | — |
| Bank 1 program (128K) (double word program) | 1.0 | 2.5 | 9.3 | s | — |
| Bank 2 program (192K) (double word program) | 1.5 | 3.7 | 14.0 | s | — |
| Bank 3 program (128K) (double word program) | 1.0 | 2.5 | 9.3 | s | — |
| Sector erase (8K) | 0.6 0.5 | 0.9 0.8 | 1.0 0.9 | s | not preprogrammed preprogrammed |
| Sector erase (32K) | 1.1 0.8 | 2.0 1.8 | 2.7 2.5 | s | not preprogrammed preprogrammed |
| Sector erase (64K) | 1.7 1.3 | 3.7 3.3 | 5.1 4.7 | s | not preprogrammed preprogrammed |
| Bank 0 erase (384K) ⁽³⁾ | 8.2 5.8 | 20.2 17.7 | 28.6 26.1 | s | not preprogrammed preprogrammed |
| Bank 1 erase (128K) ⁽³⁾ | 3.0 2.2 | 7.0 6.2 | 9.8 9.0 | s | not preprogrammed preprogrammed |
| Bank 2 erase (192K) ⁽³⁾ | 4.3 3.1 | 10.3 9.1 | 14.5 13.3 | s | not preprogrammed preprogrammed |
| Bank 3 erase (128K) ⁽³⁾ | 3.0 2.2 | 7.0 6.2 | 9.8 9.0 | s | not preprogrammed preprogrammed |
| I-Module erase (512K) ⁽⁴⁾ | 11.2 7.6 | 27.2 23.5 | 38.4 34.7 | s | not preprogrammed preprogrammed |
| X-Module erase (320K) ⁽⁴⁾ | 7.3 4.9 | 17.3 14.8 | 24.3 21.8 | s | not preprogrammed preprogrammed |
| Chip erase (832K) ⁽⁵⁾ | 18.5 12.0 | 44.4 37.9 | 62.6 56.1 | s | not preprogrammed preprogrammed |
| Recovery from power-down (t_{PD}) | — | 40 | 40 | μs | ⁽⁶⁾ |
| Program suspend latency ⁽⁶⁾ | — | 10 | 10 | μs | |

Table 91. Flash characteristics (continued)

| Parameter | Typical $T_A = 25^\circ\text{C}$ | Maximum $T_A = 125^\circ\text{C}$ | | Unit | Notes |
|---|-------------------------------------|--------------------------------------|-------------|---------------|--------------------------------|
| | 0 cycles ⁽¹⁾ | 0 cycles ⁽¹⁾ | 100k cycles | | |
| Erase suspend latency ⁽⁶⁾ | – | 30 | 30 | μs | |
| Erase suspend request Rate ⁽⁶⁾ | 20 | 20 | 20 | ms | Min delay between two requests |
| Set protection ⁽⁶⁾ | 40 | 170 | 170 | μs | |

1. The figures are given after about 100 cycles due to testing routines (0 cycles at the final customer).
2. Word and Double Word Programming times are provided as average value derived from a full sector programming time: Absolute value of a Word or Double Word Programming time could be longer than the provided average value.
3. Bank Erase is obtained through a multiple Sector Erase operation (setting bits related to all sectors of the Bank).
4. Module Erase is obtained through a sequence of two Bank Erase operations (since each module is composed by two Banks).
5. Chip Erase is obtained through a sequence of two Module Erase operations on I- and X-Module.
6. Not 100% tested, guaranteed by design characterization

Table 92. Data retention characteristics

| Number of program / erase cycles ($-40^\circ\text{C} \leq T_A \leq 125^\circ\text{C}$) | Data retention time (average ambient temperature 60°C) | |
|---|--|---|
| | 832 Kbyte (code store) | 64 Kbyte (EEPROM emulation) ⁽¹⁾ |
| 0 - 100 | > 20 years | > 20 years |
| 1000 | - | > 20 years |
| 10000 | - | 10 years |
| 100000 | - | 1 year |

1. Two 64 Kbyte Flash Sectors may be typically used to emulate up to 4, 8 or 16 Kbytes of EEPROM. Therefore, in case of an emulation of a 16 Kbyte EEPROM, 100000 Flash Program / Erase cycles are equivalent to 800000 EEPROM Program/Erase cycles.
For an efficient use of the Read While Write feature and/or EEPROM Emulation please refer to dedicated Application Note document (AN2061 - EEPROM Emulation with ST10F2xx). Contact your local field service, local sales person or STMicroelectronics representative to obtain a copy of such a guideline document.

23.7 A/D converter characteristics

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$, $4.5V \leq V_{AREF} \leq V_{DD}$,
 $V_{SS} \leq V_{AGND} \leq V_{SS} + 0.2V$

Table 93. A/D converter characteristics

| Symbol | Parameter | Test condition | Limit values | | Unit |
|----------------------|---|------------------------------------|--------------|----------------|---------------|
| | | | Min. | Max. | |
| V_{AREF} SR | Analog reference voltage ⁽¹⁾ | | 4.5 | V_{DD} | V |
| V_{AGND} SR | Analog ground voltage | | V_{SS} | $V_{SS} + 0.2$ | V |
| V_{AIN} SR | Analog Input voltage ⁽²⁾ | | V_{AGND} | V_{AREF} | V |
| I_{AREF} CC | Reference supply current | Running mode | – | 5 | mA |
| | | ⁽³⁾ Power Down mode | – | 1 | μA |
| t_S CC | Sample time | ⁽⁴⁾ | 1 | – | μs |
| t_C CC | Conversion time | ⁽⁵⁾ | 3 | – | μs |
| DNL CC | Differential nonlinearity ⁽⁶⁾ | No overload | –1 | +1 | LSB |
| INL CC | Integral nonlinearity ⁽⁶⁾ | No overload | –1.5 | +1.5 | LSB |
| OFS CC | Offset error ⁽⁶⁾ | No overload | –1.5 | +1.5 | LSB |
| TUE CC | Total unadjusted error ⁽⁶⁾ | Port5 | –2.0 | +2.0 | LSB |
| | | Port1 - No overload ⁽³⁾ | –5.0 | +5.0 | LSB |
| | | Port1 - Overload ⁽³⁾ | –7.0 | +7.0 | LSB |
| K CC | Coupling factor between inputs ^{(3) (7)} | On both Port5 and Port1 | – | 10^{-6} | – |
| C_{P1} CC | Input pin capacitance ^{(3) (8)} | | – | 3 | pF |
| C_{P2} CC | | Port5 Port1 | – | 4 6 | pF pF |
| C_S CC | Sampling capacitance ⁽³⁾⁽⁸⁾ | | – | 3.5 | pF |
| R_{SW} CC | Analog switch resistance ^{(3) (8)} | Port5 | – | 600 | W |
| | | Port1 | – | 1600 | W |
| R_{AD} CC | | | – | 1300 | W |

- V_{AREF} can be tied to ground when A/D converter is not in use: An extra consumption (around 200 μA) on main V_{DD} is added due to internal analog circuitry not completely turned off. Therefore, it is suggested to maintain the V_{AREF} at V_{DD} level even when not in use, and eventually switch off the A/D converter circuitry setting bit ADOFF in ADCON register.
- V_{AIN} may exceed V_{AGND} or V_{AREF} up to the absolute maximum ratings. However, the conversion result in these cases will be 0x000_H or 0x3FF_H, respectively.
- Not 100% tested, guaranteed by design characterization.
- During the sample time, the input capacitance C_{AIN} can be charged/discharged by the external source. The internal resistance of the analog source must allow the capacitance to reach its final voltage level within t_S . After the end of the sample time t_S , changes of the analog input voltage have no effect on the conversion result.
Values for the sample clock t_S depend on programming and can be taken from [Table 94](#).
- This parameter includes the sample time t_S , the time for determining the digital result and the time to load the result register with the conversion result. Values for the conversion clock t_{CC} depend on programming and can be taken from next [Table 94](#).

6. DNL, INL, OFS and TUE are tested at $V_{AREF} = 5.0V$, $V_{AGND} = 0V$, $V_{DD} = 5.0V$. It is guaranteed by design characterization for all other voltages within the defined voltage range.
 "LSB" has a value of $V_{AREF}/1024$.
 For Port5 channels, the specified TUE ($\pm 2LSB$) is also guaranteed with an overload condition (see I_{OV} specification) occurring on a maximum of 2 not selected analog input pins of Port5 and the absolute sum of input overload currents on all Port5 analog input pins does not exceed 10 mA.
 For Port1 channels, the specified TUE is guaranteed when no overload condition is applied to Port1 pins: When an overload condition occurs on a maximum of 2 not selected analog input pins of Port1 and the input positive overload current on all analog input pins does not exceed 10 mA (either dynamic or static injection), the specified TUE is degraded ($\pm 7LSB$). To obtain the same accuracy, the negative injection current on Port1 pins shall not exceed -1mA in case of both dynamic and static injection.
7. The coupling factor is measured on a channel while an overload condition occurs on the adjacent not selected channels with the overload current within the different specified ranges (for both positive and negative injection current).
8. Refer to scheme shown in [Figure 47](#)

23.7.1 Conversion timing control

When a conversion starts, first the capacitances of the converter are loaded via the respective analog input pin to the current analog input voltage. The time to load the capacitances is referred to as sample time. Next, the sampled voltage is converted in several successive steps into a digital value, which corresponds to the 10-bit resolution of the ADC. During these steps the internal capacitances are repeatedly charged and discharged via the V_{AREF} pin.

The current that must be drawn from the sources for sampling and changing charges depends on the duration of each step because the capacitors must reach their final voltage level within the given time, at least with a certain approximation. However, the maximum current that a source can deliver depends on its internal resistance.

The time that the two different actions take during conversion (sampling and converting) can be programmed within a certain range in the ST10F276 relative to the CPU clock. The absolute time consumed by the different conversion steps is therefore independent from the general speed of the controller. This allows adjusting the ST10F276 A/D converter to the properties of the system:

Fast conversion can be achieved by programming the respective times to their absolute possible minimum. This is preferable for scanning high frequency signals. However, the internal resistance of analog source and analog supply must be sufficiently low.

High internal resistance can be achieved by programming the respective times to a higher value or to the possible maximum. This is preferable when using analog sources and supply with a high internal resistance in order to keep the current as low as possible. However, the conversion rate in this case may be considerably lower.

The conversion times are programmed via the upper 4 bits of register ADCON. Bit fields ADCTC and ADSTC define the basic conversion time and in particular the partition between the sample phase and comparison phases. The table below lists the possible combinations. The timings refer to the unit TCL, where $f_{CPU} = 1/2TCL$. A complete conversion time includes the conversion itself, the sample time and the time required to transfer the digital value to the result register.

Table 94. A/D Converter programming

| ADCTC | ADSTC | Sample | Comparison | Extra | Total conversion |
|-------|-------|-----------|------------|----------|------------------|
| 00 | 00 | TCL * 120 | TCL * 240 | TCL * 28 | TCL * 388 |
| 00 | 01 | TCL * 140 | TCL * 280 | TCL * 16 | TCL * 436 |

Table 94. A/D Converter programming (continued)

| ADCTC | ADSTC | Sample | Comparison | Extra | Total conversion |
|-------|-------|------------|------------|-----------|------------------|
| 00 | 10 | TCL * 200 | TCL * 280 | TCL * 52 | TCL * 532 |
| 00 | 11 | TCL * 400 | TCL * 280 | TCL * 44 | TCL * 724 |
| 11 | 00 | TCL * 240 | TCL * 480 | TCL * 52 | TCL * 772 |
| 11 | 01 | TCL * 280 | TCL * 560 | TCL * 28 | TCL * 868 |
| 11 | 10 | TCL * 400 | TCL * 560 | TCL * 100 | TCL * 1060 |
| 11 | 11 | TCL * 800 | TCL * 560 | TCL * 52 | TCL * 1444 |
| 10 | 00 | TCL * 480 | TCL * 960 | TCL * 100 | TCL * 1540 |
| 10 | 01 | TCL * 560 | TCL * 1120 | TCL * 52 | TCL * 1732 |
| 10 | 10 | TCL * 800 | TCL * 1120 | TCL * 196 | TCL * 2116 |
| 10 | 11 | TCL * 1600 | TCL * 1120 | TCL * 164 | TCL * 2884 |

Note: The total conversion time is compatible with the formula valid for ST10F269, while the meaning of the bit fields ADCTC and ADSTC is no longer compatible: The minimum conversion time is 388 TCL, which at 40 MHz CPU frequency corresponds to 4.85µs (see ST10F269).

23.7.2 A/D conversion accuracy

The A/D converter compares the analog voltage sampled on the selected analog input channel to its analog reference voltage (V_{AREF}) and converts it into 10-bit digital data. The absolute accuracy of the A/D conversion is the deviation between the input analog value and the output digital value. It includes the following errors:

- Offset error (OFS)
- Gain error (GE)
- Quantization error
- Nonlinearity error (differential and integral)

These four error quantities are explained below using [Figure 46](#).

Offset error

Offset error is the deviation between actual and ideal A/D conversion characteristics when the digital output value changes from the minimum (zero voltage) 00 to 01 (Figure 46, see OFS).

Gain error

Gain error is the deviation between the actual and ideal A/D conversion characteristics when the digital output value changes from the 3FE to the maximum 3FF, once offset error is subtracted. Gain error combined with offset error represents the so-called full-scale error (Figure 46, OFS + GE).

Quantization error

Quantization error is the intrinsic error of the A/D converter and is expressed as 1/2 LSB.

Nonlinearity error

Nonlinearity error is the deviation between actual and the best-fitting A/D conversion characteristics (see Figure 46):

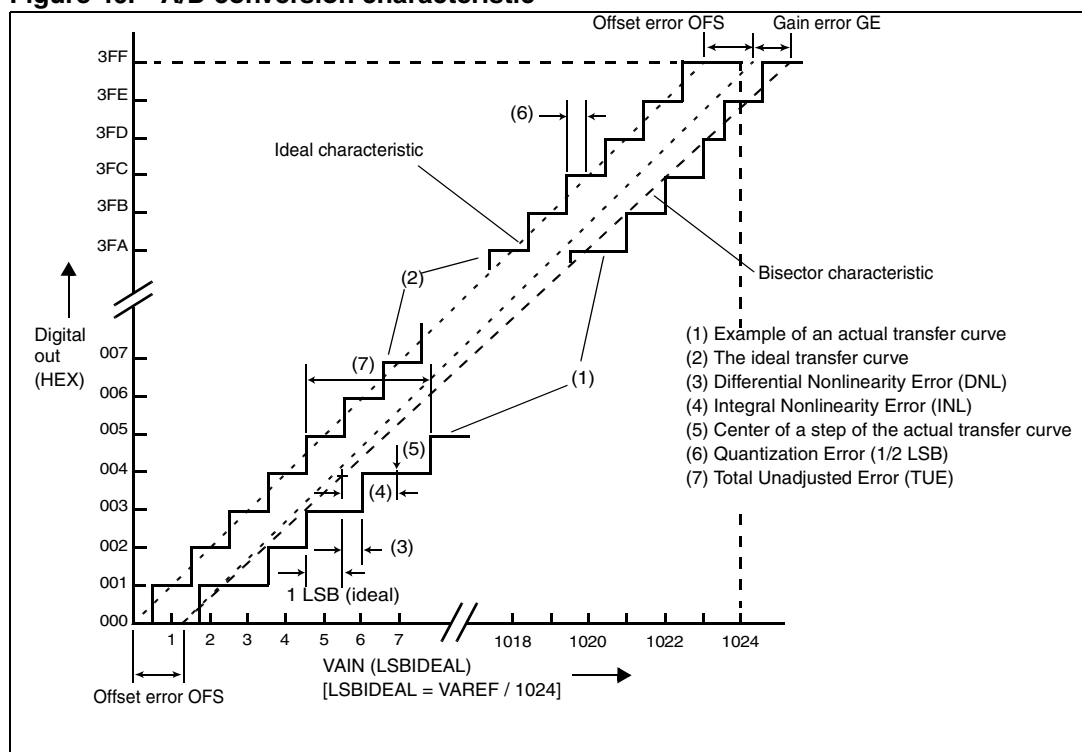
- Differential nonlinearity error is the actual step dimension versus the ideal one ($1 \text{ LSB}_{\text{IDEAL}}$).
- Integral nonlinearity error is the distance between the center of the actual step and the center of the bisector line, in the actual characteristics. Note that for integral nonlinearity error, the effect of offset, gain and quantization errors is not included.

Note: Bisector characteristic is obtained drawing a line from $1/2 \text{ LSB}$ before the first step of the real characteristic, and $1/2 \text{ LSB}$ after the last step again of the real characteristic.

23.7.3 Total unadjusted error

The total unadjusted error (TUE) specifies the maximum deviation from the ideal characteristic: The number provided in the datasheet represents the maximum error with respect to the entire characteristic. It is a combination of the offset, gain and integral linearity errors. The different errors may compensate each other depending on the relative sign of the offset and gain errors. Refer to Figure 46, see TUE.

Figure 46. A/D conversion characteristic



23.7.4 Analog reference pins

The accuracy of the A/D converter depends on the accuracy of its analog reference: A noise in the reference results in proportionate error in a conversion. A low pass filter on the A/D converter reference source (supplied through pins V_{AREF} and V_{AGND}), is recommended in order to clean the signal, minimizing the noise. A simple capacitive bypassing may be sufficient in most cases; in presence of high RF noise energy, inductors or ferrite beads may be necessary.

In this architecture, V_{AREF} and V_{AGND} pins also represent the power supply of the analog circuitry of the A/D converter: There is an effective DC current requirement from the reference voltage by the internal resistor string in the R-C DAC array and by the rest of the analog circuitry.

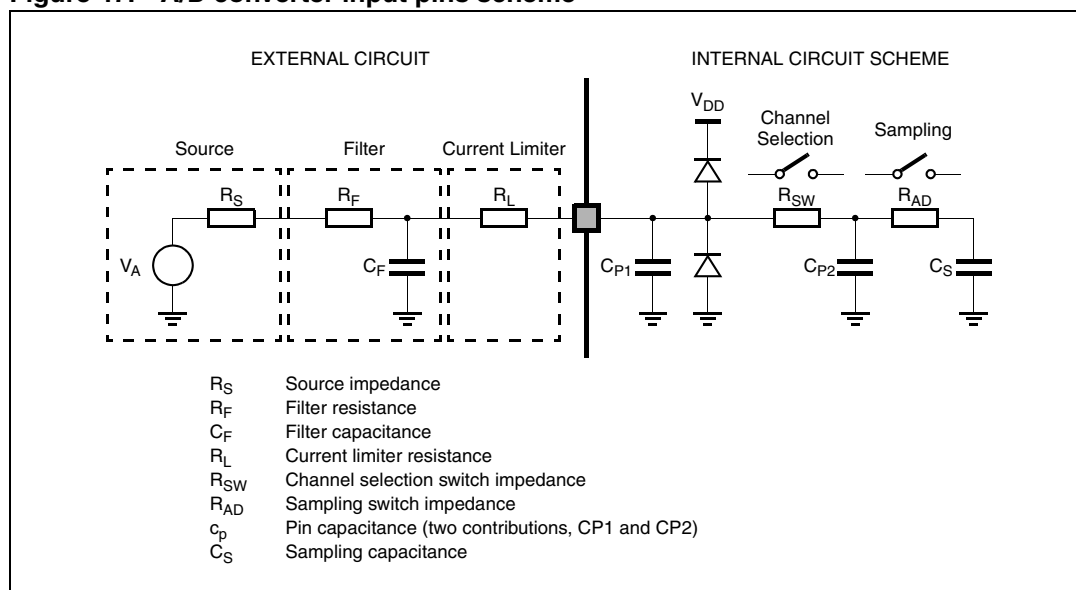
An external resistance on V_{AREF} could introduce error under certain conditions: For this reason, series resistance is not advisable and more generally, any series devices in the filter network should be designed to minimize the DC resistance.

23.7.5 Analog input pins

To improve the accuracy of the A/D converter, analog input pins must have low AC impedance. Placing a capacitor with good high frequency characteristics at the input pin of the device can be effective: The capacitor should be as large as possible, ideally infinite. This capacitor contributes to attenuating the noise present on the input pin; moreover, its source charges during the sampling phase, when the analog signal source is a high-impedance source.

A real filter is typically obtained by using a series resistance with a capacitor on the input pin (simple RC Filter). The RC filtering may be limited according to the value of source impedance of the transducer or circuit supplying the analog signal to be measured. The filter at the input pins must be designed taking into account the dynamic characteristics of the input signal (bandwidth).

Figure 47. A/D converter input pins scheme



Input leakage and external circuit

The series resistor utilized to limit the current to a pin (see R_L in Figure 47), in combination with a large source impedance, can lead to a degradation of A/D converter accuracy when input leakage is present.

Data about maximum input leakage current at each pin is provided in the datasheet (Electrical Characteristics section). Input leakage is greatest at high operating temperatures and in general decreases by one half for each 10°C decrease in temperature.

Considering that, for a 10-bit A/D converter one count is about 5mV (assuming $V_{AREF} = 5\text{V}$), an input leakage of 100nA acting through an $R_L = 50\text{k}\Omega$ of external resistance leads to an error of exactly one count (5mV); if the resistance were 100k Ω , the error would become two counts.

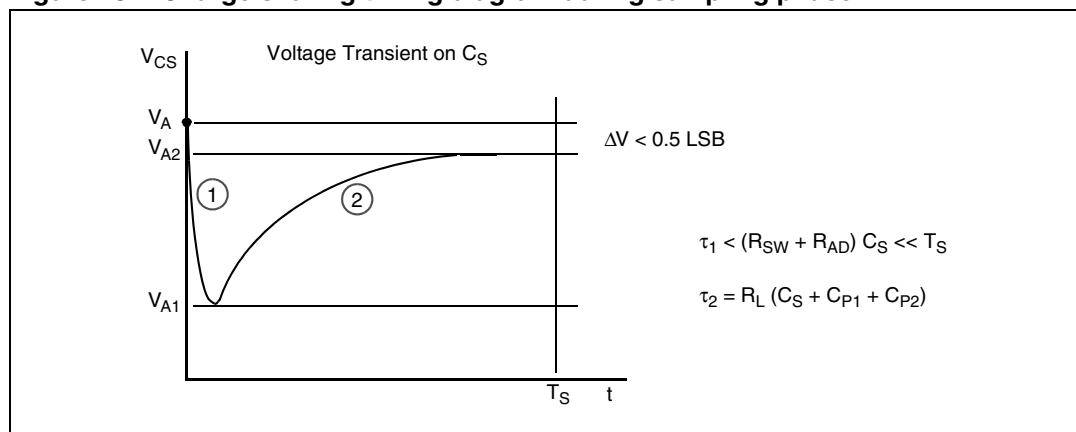
Eventual additional leakage due to external clamping diodes must also be taken into account in computing the total leakage affecting the A/D converter measurements. Another contribution to the total leakage is represented by the charge sharing effects with the sampling capacitance: C_S being substantially a switched capacitance, with a frequency equal to the conversion rate of a single channel (maximum when fixed channel continuous conversion mode is selected), it can be seen as a resistive path to ground. For instance, assuming a conversion rate of 250 kHz, with C_S equal to 4pF, a resistance of 1M Ω is obtained ($R_{EQ} = 1 / f_C C_S$, where f_C represents the conversion rate at the considered channel). To minimize the error induced by the voltage partitioning between this resistance (sampled voltage on C_S) and the sum of $R_S + R_F + R_L + R_{SW} + R_{AD}$, the external circuit must be designed to respect the following relation:

$$V_A \cdot \frac{R_S + R_F + R_L + R_{SW} + R_{AD}}{R_{EQ}} < \frac{1}{2} \text{ LSB}$$

The formula above places constraints on external network design, in particular on resistive path.

A second aspect involving the capacitance network must be considered. Assuming the three capacitances C_F , C_{P1} and C_{P2} are initially charged at the source voltage V_A (refer to the equivalent circuit shown in Figure 47), when the sampling phase is started (A/D switch close), a charge sharing phenomena is installed.

Figure 48. Charge sharing timing diagram during sampling phase



In particular two different transient periods can be distinguished (see Figure 48):

1. A first and quick charge transfer from the internal capacitances C_{P1} and C_{P2} to the sampling capacitance C_S occurs (C_S is supposed initially completely discharged): Considering a worst case (since the time constant in reality would be faster) in which C_{P2} is reported in parallel to C_{P1} (call $C_P = C_{P1} + C_{P2}$), the two capacitances C_P and C_S are in series and the time constant is:

$$\tau_1 = (R_{SW} + R_{AD}) \cdot \frac{C_P \cdot C_S}{C_P + C_S}$$

This relation can again be simplified considering only C_S as an additional worst condition. In reality, the transient is faster, but the A/D converter circuitry has been designed to also be robust in the very worst case: The sampling time T_S is always much longer than the internal time constant:

$$\tau_1 < (R_{SW} + R_{AD}) \cdot C_S \ll T_S$$

The charge of C_{P1} and C_{P2} is also redistributed on C_S , determining a new value of the voltage V_{A1} on the capacitance according to the following equation:

$$V_{A1} \cdot (C_S + C_{P1} + C_{P2}) = V_A \cdot (C_{P1} + C_{P2})$$

2. A second charge transfer also involves C_F (that is typically bigger than the on-chip capacitance) through the resistance R_L : Again considering the worst case in which C_{P2} and C_S were in parallel to C_{P1} (since the time constant in reality would be faster), the time constant is:

$$\tau_2 < R_L \cdot (C_S + C_{P1} + C_{P2})$$

In this case, the time constant depends on the external circuit: In particular, imposing that the transient is completed well before the end of sampling time T_S , a constraint on R_L sizing is obtained:

$$10 \cdot \tau_2 = 10 \cdot R_L \cdot (C_S + C_{P1} + C_{P2}) \leq T_S$$

Of course, R_L must also be sized according to the current limitation constraints, in combination with R_S (source impedance) and R_F (filter resistance). Being that C_F is definitely bigger than C_{P1} , C_{P2} and C_S , then the final voltage V_{A2} (at the end of the charge transfer transient) will be much higher than V_{A1} . The following equation must be respected (charge balance assuming now C_S already charged at V_{A1}):

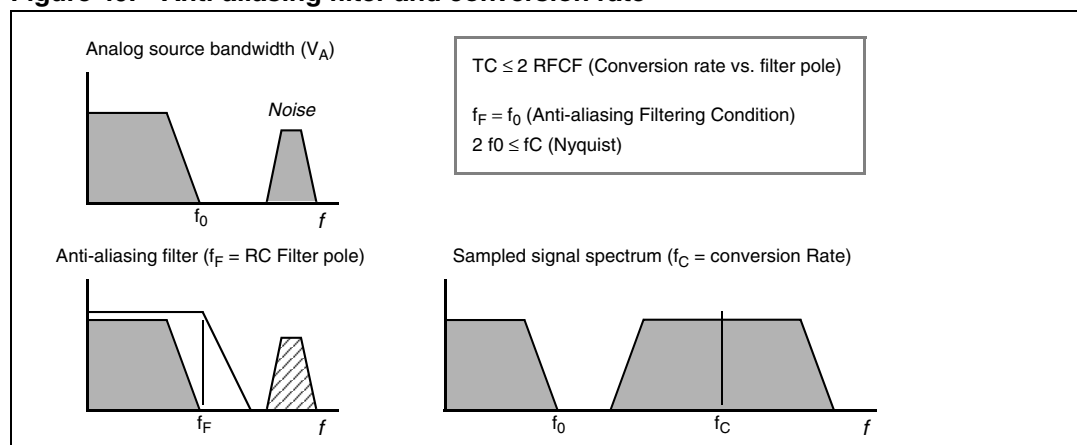
$$V_{A2}(C_S + C_{P1} + C_{P2} + C_F) = V_A C_F + V_{A1}(C_{P1} + C_{P2} + C_S)$$

The two transients above are not influenced by the voltage source that, due to the presence of the $R_F C_F$ filter, cannot provide the extra charge to compensate for the voltage drop on C_S with respect to the ideal source V_A ; the time constant $R_F C_F$ of the filter is very high with respect to the sampling time (T_S). The filter is typically designed to act as anti-aliasing (see Figure 49).

Calling f_0 the bandwidth of the source signal (and as a consequence the cut-off frequency of the anti-aliasing filter, f_F), according to Nyquist theorem the conversion rate f_C must be at least $2f_0$, meaning that the constant time of the filter is greater than or at least equal to twice the conversion period (T_C). Again the conversion period T_C is longer than the sampling time T_S , which is just a portion of it, even when fixed channel continuous conversion mode is selected (fastest conversion rate at a specific channel): In conclusion, it is evident that the

time constant of the filter $R_F C_F$ is definitely much higher than the sampling time T_S , so the charge level on C_S cannot be modified by the analog signal source during the time in which the sampling switch is closed.

Figure 49. Anti-aliasing filter and conversion rate



The considerations above lead to impose new constraints to the external circuit, to reduce the accuracy error due to the voltage drop on C_S ; from the two charge balance equations above, it is simple to derive the following relation between the ideal and real sampled voltage on C_S :

$$\frac{V_A}{V_{A2}} = \frac{C_{P1} + C_{P2} + C_F}{C_{P1} + C_{P2} + C_F + C_S}$$

From this formula, in the worst case (when V_A is maximum, that is for instance 5V), assuming to accept a maximum error of half a count ($\sim 2.44\text{mV}$), it is immediately evident that a constraint is on C_F value:

$$C_F > 2048 C_S$$

The next section provides an example of how to design the external network, based on some reasonable values for the internal parameters and on a hypothesis on the characteristics of the analog signal to be sampled.

23.7.6 Example of external network sizing

The following hypothesis is formulated in order to proceed with designing the external network on A/D converter input pins:

- Analog signal source bandwidth (f_0): 10 kHz
- Conversion rate (f_C): 25 kHz
- Sampling time (T_S): 1 μ s
- Pin input capacitance (C_{P1}): 5 pF
- Pin input routing capacitance (C_{P2}): 1 pF
- Sampling capacitance (C_S): 4 pF
- Maximum input current injection (I_{INJ}): 3 mA
- Maximum analog source voltage (V_{AM}): 12 V
- Analog source impedance (R_S): 100 Ω
- Channel switch resistance (R_{SW}): 500 Ω
- Sampling switch resistance (R_{AD}): 200 Ω

1. Supposing to design the filter with the pole exactly at the maximum frequency of the signal, the time constant of the filter is:

$$R_C C_F = \frac{1}{2\pi f_0} = 15.9 \mu s$$

2. Using the relation between C_F and C_S and taking some margin (4000 instead of 2048), it is possible to define C_F :

$$C_F = 4000 C_S = 16 nF$$

3. As a consequence of Step 1 and 2, RC can be chosen:

$$R_F = \frac{1}{2\pi f_0 C_F} = 995 \Omega \approx 1 k\Omega$$

4. Considering the current injection limitation and supposing that the source can go up to 12V, the total series resistance can be defined as:

$$R_S + R_F + R_L = \frac{V_{AM}}{I_{INJ}} = 4 k\Omega$$

from which is now simple to define the value of R_L :

$$R_L = \frac{V_{AM}}{I_{INJ}} - R_F - R_S = 2.9 k\Omega$$

Now, the three elements of the external circuit R_F , C_F and R_L are defined. Some conditions discussed in the previous paragraphs have been used to size the component; the others must now be verified. The relation which allows to minimize the accuracy error introduced by the switched capacitance equivalent resistance is in this case:

$$R_{EQ} = \frac{1}{f_C C_S} = 10 M\Omega$$

So the error due to the voltage partitioning between the real resistive path and C_S is less

then half a count (considering the worst case when $V_A = 5V$):

$$V_A \frac{R_S + R_F + R_L + R_{SW} + R_{AD}}{R_{EQ}} = 2.35mV < \frac{1}{2}LSB$$

The other conditions to verify are if the time constants of the transients are really and significantly shorter than the sampling period duration T_S :

$$\tau_1 = (R_{SW} + R_{AD}) \cdot C_S = 2.8ns \ll T_S = 1\mu s$$

$$10\tau_2 = 10R_L(C_S + C_{P1} + C_{P2}) = 290ns < T_S = 1\mu s$$

For a complete set of parameters characterizing the ST10F276 A/D converter equivalent circuit, refer to A/D Converter Characteristics table at page 185.

23.8 AC characteristics

23.8.1 Test waveforms

Figure 50. Input/output waveforms

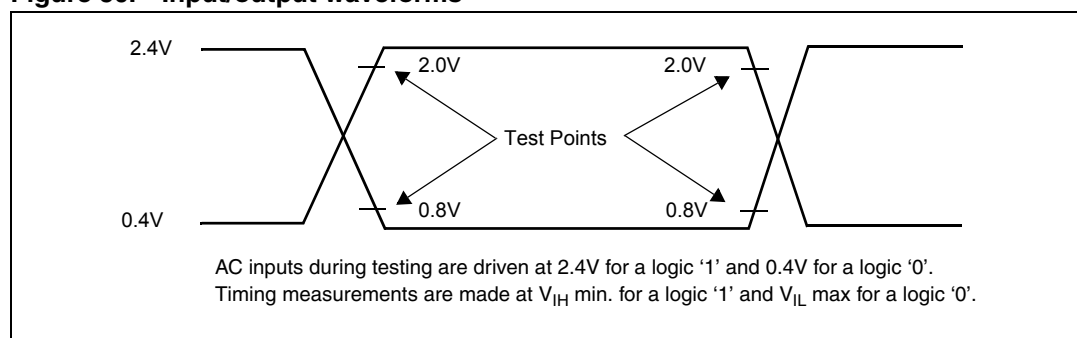
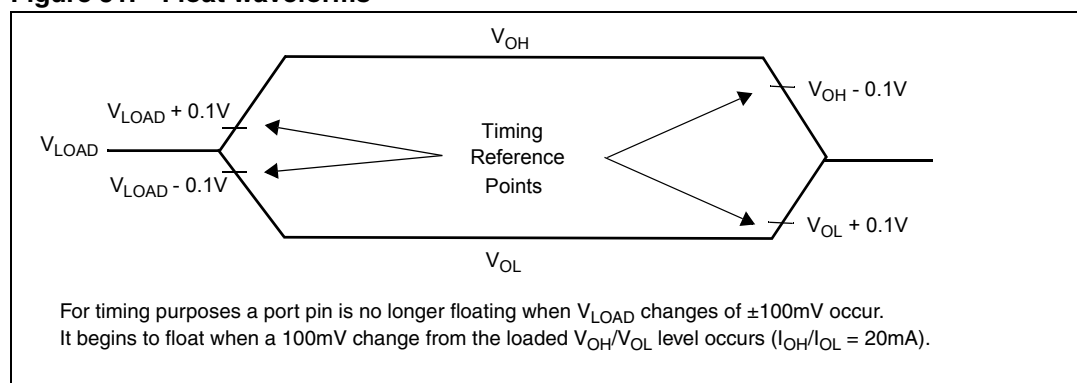


Figure 51. Float waveforms



23.8.2 Definition of internal timing

The internal operation of the ST10F276 is controlled by the internal CPU clock f_{CPU} . Both edges of the CPU clock can trigger internal (for example pipeline) or external (for example bus cycles) operations.

The specification of the external timing (AC Characteristics) therefore depends on the time between two consecutive edges of the CPU clock, called "TCL".

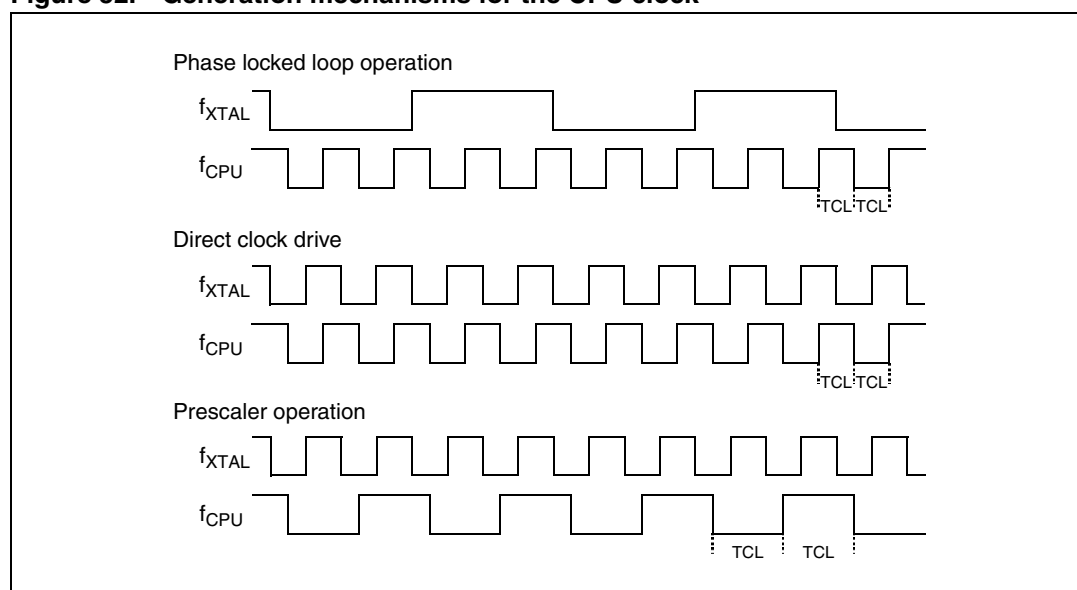
The CPU clock signal can be generated by different mechanisms. The duration of TCL and its variation (and also the derived external timing) depends on the mechanism used to generate f_{CPU} .

This influence must be regarded when calculating the timings for the ST10F276.

The example for PLL operation shown in Figure 52 refers to a PLL factor of 4.

The mechanism used to generate the CPU clock is selected during reset by the logic levels on pins P0.15-13 (POH.7-5).

Figure 52. Generation mechanisms for the CPU clock



23.8.3 Clock generation modes

The following table associates the combinations of these 3 bits with the respective clock generation mode.

Table 95. On-chip clock generator selections

| P0.15-13 (P0H.7-5) | CPU frequency $f_{\text{CPU}} = f_{\text{XTAL}} \times F$ | External clock input range ⁽¹⁾⁽²⁾ | Notes |
|-----------------------|--|---|---|
| 1 1 1 | $F_{\text{XTAL}} \times 4$ | 4 to 8 MHz | Default configuration |
| 1 1 0 | $F_{\text{XTAL}} \times 3$ | 5.3 to 10.6 MHz | |
| 1 0 1 | $F_{\text{XTAL}} \times 8$ | 4 to 8 MHz | |
| 1 0 0 | $F_{\text{XTAL}} \times 5$ | 6.4 to 12 MHz | |
| 0 1 1 | $F_{\text{XTAL}} \times 1$ | 1 to 64 MHz | Direct Drive (oscillator bypassed) ⁽³⁾ |
| 0 1 0 | $F_{\text{XTAL}} \times 10$ | 4 to 6.4 MHz | |
| 0 0 1 | $F_{\text{XTAL}} / 2$ | 4 to 12 MHz | CPU clock via prescaler ⁽³⁾ |
| 0 0 0 | $F_{\text{XTAL}} \times 16$ | 4 MHz | |

1. The external clock input range refers to a CPU clock range of 1...64 MHz. Moreover, the PLL usage is limited to 4-12 MHz input frequency range. All configurations need a crystal (or ceramic resonator) to generate the CPU clock through the internal oscillator amplifier (apart from Direct Drive); on the contrary, the clock can be forced through an external clock source only in Direct Drive mode (on-chip oscillator amplifier disabled, so no crystal or resonator can be used).
2. The limits on input frequency are 4-12 MHz since the usage of the internal oscillator amplifier is required. Also, when the PLL is not used and the CPU clock corresponds to $F_{\text{XTAL}}/2$, an external crystal or resonator must be used: It is not possible to force any clock through an external clock source.
3. The maximum depends on the duty cycle of the external clock signal: When 64 MHz is used, 50% duty cycle shall be granted (low phase = high phase = 7.8ns); when 32 MHz is selected, a 25% duty cycle can be accepted (minimum phase, high or low, again equal to 7.8ns).

23.8.4 Prescaler operation

When pins P0.15-13 (P0H.7-5) equal '001' during reset, the CPU clock is derived from the internal oscillator (input clock signal) by a 2:1 prescaler.

The frequency of f_{CPU} is half the frequency of f_{XTAL} and the high and low time of f_{CPU} (that is, the duration of an individual TCL) is defined by the period of the input clock f_{XTAL} .

The timings listed in the AC Characteristics that refer to TCL can therefore be calculated using the period of f_{XTAL} for any TCL.

Note that if the bit OWDDIS in SYSCON register is cleared, the PLL runs on its free-running frequency and delivers the clock signal for the Oscillator Watchdog. If bit OWDDIS is set, then the PLL is switched off.

23.8.5 Direct drive

When pins P0.15-13 (P0H.7-5) equal '011' during reset, the on-chip phase locked loop is disabled, the on-chip oscillator amplifier is bypassed and the CPU clock is directly driven by the input clock signal on XTAL1 pin.

The frequency of the CPU clock (f_{CPU}) directly follows the frequency of f_{XTAL} so the high and low time of f_{CPU} (that is, the duration of an individual TCL) is defined by the duty cycle of the input clock f_{XTAL} .

Therefore, the timings given in this chapter refer to the minimum TCL. This minimum value can be calculated by the following formula:

$$TCL_{min} = 1/f_{XTAL} \times DC_{min}$$

DC = duty cycle

For two consecutive TCLs, the deviation caused by the duty cycle of f_{XTAL} is compensated, so the duration of 2TCL is always $1/f_{XTAL}$.

The minimum value TCL_{min} is used only once for timings that require an odd number of TCLs (1, 3, ...). Timings that require an even number of TCLs (2, 4, ...) may use the formula:

$$2TCL = 1/f_{XTAL}$$

The address float timings in multiplexed bus mode (t_{11} and t_{45}) use the maximum duration of TCL ($TCL_{max} = 1/f_{XTAL} \times DC_{max}$) instead of TCL_{min} .

Similarly to what happens for Prescaler Operation, if the bit OWDDIS in SYSCON register is cleared, the PLL runs on its free-running frequency and delivers the clock signal for the Oscillator Watchdog. If bit OWDDIS is set, then the PLL is switched off.

23.8.6 Oscillator watchdog (OWD)

An on-chip watchdog oscillator is implemented in the ST10F276. This feature is used for safety operation with an external crystal oscillator (available only when using direct drive mode with or without prescaler, so the PLL is not used to generate the CPU clock multiplying the frequency of the external crystal oscillator). This watchdog oscillator operates as following.

The reset default configuration enables the watchdog oscillator. It can be disabled by setting the OWDDIS (bit 4) of SYSCON register.

When the OWD is enabled, the PLL runs at its free-running frequency and it increments the watchdog counter. On each transition of external clock, the watchdog counter is cleared. If an external clock failure occurs, then the watchdog counter overflows (after 16 PLL clock cycles).

The CPU clock signal is switched to the PLL free-running clock signal and the oscillator watchdog Interrupt Request is flagged. The CPU clock will not switch back to the external clock even if a valid external clock exists on XTAL1 pin. Only a hardware reset (or bidirectional Software / Watchdog reset) can switch the CPU clock source back to direct clock input.

When the OWD is disabled, the CPU clock is always the external oscillator clock (in Direct Drive or Prescaler Operation) and the PLL is switched off to decrease consumption supply current.

23.8.7 Phase locked loop (PLL)

For all other combinations of pins P0.15-13 (P0H.7-5) during reset the on-chip phase locked loop is enabled and it provides the CPU clock (see [Table 95](#)). The PLL multiplies the input frequency by the factor F which is selected via the combination of pins P0.15-13 ($f_{CPU} = f_{XTAL} \times F$). With every F'th transition of f_{XTAL} the PLL circuit synchronizes the CPU clock to the input clock. This synchronization is done smoothly, so the CPU clock frequency does not change abruptly.

Due to this adaptation to the input clock, the frequency of f_{CPU} is constantly adjusted so it is locked to f_{XTAL} . The slight variation causes a jitter of f_{CPU} which also effects the duration of individual TCLs.

The timings listed in the AC Characteristics that refer to TCLs therefore must be calculated using the minimum TCL that is possible under the respective circumstances.

The real minimum value for TCL depends on the jitter of the PLL. The PLL tunes f_{CPU} to keep it locked on f_{XTAL} . The relative deviation of TCL is the maximum when it is referred to one TCL period.

This is especially important for bus cycles using wait states and e.g. for the operation of timers, serial interfaces, etc. For all slower operations and longer periods (such as, for example, pulse train generation or measurement, lower baud rates) the deviation caused by the PLL jitter is negligible. Refer to next [Section 23.8.9: PLL Jitter](#) for more details.

23.8.8 Voltage controlled oscillator

The ST10F276 implements a PLL which combines different levels of frequency dividers with a Voltage Controlled Oscillator (VCO) working as frequency multiplier. The following table presents a detailed summary of the internal settings and VCO frequency.

Table 96. Internal PLL divider mechanism

| P0.15-13 (P0H.7-5) | XTAL frequency | Input prescaler | PLL | | Output prescaler | CPU frequency $f_{\text{CPU}} = f_{\text{XTAL}} \times F$ |
|-----------------------|-------------------|-----------------------|--------------|-----------|----------------------|--|
| | | | Multiply by | Divide by | | |
| 1 1 1 | 4 to 8 MHz | $F_{\text{XTAL}} / 4$ | 64 | 4 | — | $F_{\text{XTAL}} \times 4$ |
| 1 1 0 | 5.3 to 10.6 MHz | $F_{\text{XTAL}} / 4$ | 48 | 4 | — | $F_{\text{XTAL}} \times 3$ |
| 1 0 1 | 4 to 8 MHz | $F_{\text{XTAL}} / 4$ | 64 | 2 | — | $F_{\text{XTAL}} \times 8$ |
| 1 0 0 | 6.4 to 12 MHz | $F_{\text{XTAL}} / 4$ | 40 | 2 | — | $F_{\text{XTAL}} \times 5$ |
| 0 1 1 | 1 to 64 MHz | — | PLL bypassed | | — | $F_{\text{XTAL}} \times 1$ |
| 0 1 0 | 4 to 6.4 MHz | $F_{\text{XTAL}} / 2$ | 40 | 2 | — | $F_{\text{XTAL}} \times 10$ |
| 0 0 1 | 4 to 12 MHz | — | PLL bypassed | | $F_{\text{PLL}} / 2$ | $F_{\text{XTAL}} / 2$ |
| 0 0 0 | 4 MHz | $F_{\text{XTAL}} / 2$ | 64 | 2 | — | $F_{\text{XTAL}} \times 16$ |

The PLL input frequency range is limited to 1 to 3.5 MHz, while the VCO oscillation range is 64 to 128 MHz. The CPU clock frequency range when PLL is used is 16 to 64 MHz.

Example 1

- $F_{\text{XTAL}} = 4 \text{ MHz}$
- P0(15:13) = '110' (multiplication by 3)
- PLL input frequency = 1 MHz
- VCO frequency = 48 MHz
- PLL output frequency = 12 MHz
(VCO frequency divided by 4)
- $F_{\text{CPU}} = 12 \text{ MHz}$ (no effect of output prescaler)

Example 2

- $F_{XTAL} = 8 \text{ MHz}$
- $P0(15:13) = '100'$ (multiplication by 5)
- PLL input frequency = 2 MHz
- VCO frequency = 80 MHz
- PLL output frequency = 40 MHz (VCO frequency divided by 2)
- $F_{CPU} = 40 \text{ MHz}$ (no effect of output prescaler)

23.8.9 PLL Jitter

Two kinds of PLL jitter are defined:

- **Self referred single period jitter**
Also called "Period Jitter", it can be defined as the difference of the T_{max} and T_{min} , where T_{max} is the maximum time period of the PLL output clock and T_{min} is the minimum time period of the PLL output clock.
- **Self referred long term jitter**
Also called "N period jitter", it can be defined as the difference of T_{max} and T_{min} , where T_{max} is the maximum time difference between $N + 1$ clock rising edges and T_{min} is the minimum time difference between $N + 1$ clock rising edges. Here N should be kept sufficiently large to have the long term jitter. For $N = 1$, this becomes the single period jitter.

Jitter at the PLL output is caused by:

- Jitter in the input clock
- Noise in the PLL loop

23.8.10 Jitter in the input clock

PLL acts like a low pass filter for any jitter in the input clock. Input Clock jitter with the frequencies within the PLL loop bandwidth is passed to the PLL output and higher frequency jitter (frequency > PLL bandwidth) is attenuated at 20dB/decade.

23.8.11 Noise in the PLL loop

This condition again is attributed to the following sources:

- Device noise of the circuit in the PLL
- Noise in supply and substrate

Device noise of the circuit in the PLL

Long term jitter is inversely proportional to the bandwidth of the PLL: The wider the loop bandwidth, the lower the jitter due to noise in the loop. Moreover, long term jitter is practically independent of the multiplication factor.

The most noise sensitive circuit in the PLL circuit is definitely the VCO (Voltage Controlled Oscillator). There are two main sources of noise: Thermal (random noise, frequency independent thus practically white noise) and flicker (low frequency noise, $1/f$). For the frequency characteristics of the VCO circuitry, the effect of the thermal noise results in a $1/f^2$ region in the output noise spectrum, while the flicker noise in a $1/f^3$. Assuming a noiseless PLL input and supposing that the VCO is dominated by its $1/f^2$ noise, the R.M.S. value of the accumulated jitter is *proportional to the square root of N*, where N is the number of clock

periods within the considered time interval.

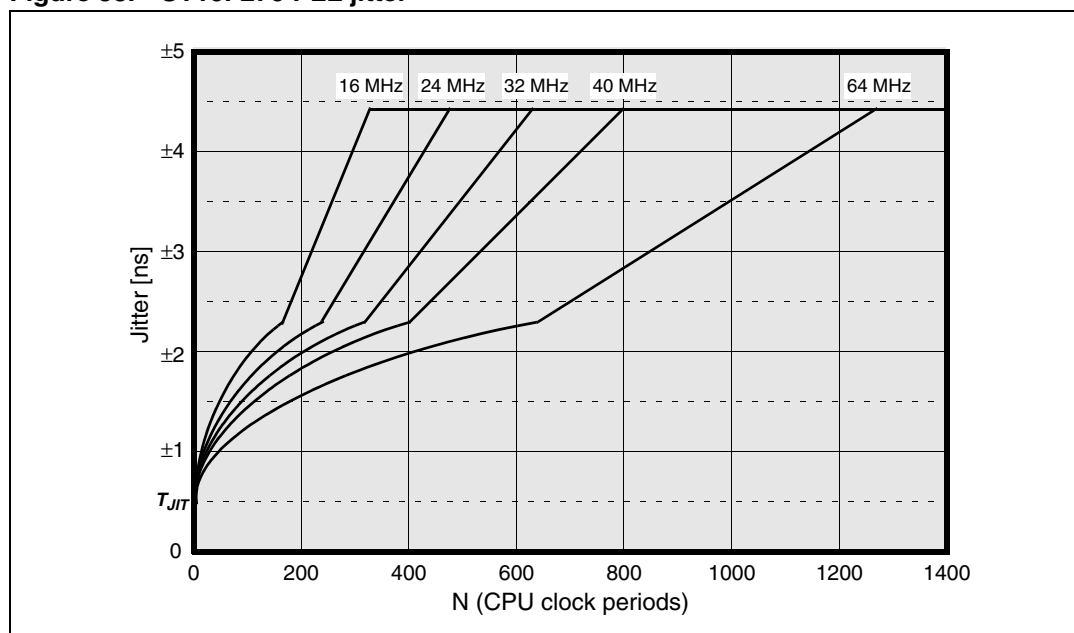
On the contrary, assuming again a noiseless PLL input and supposing that the VCO is dominated by its $1/f^3$ noise, the R.M.S. value of the accumulated jitter is *proportional to N*, where N is the number of clock periods within the considered time interval.

The jitter in the PLL loop can be modeled as dominated by the $1/f^2$ noise for N smaller than a certain value depending on the PLL output frequency and on the bandwidth characteristics of loop. Above this first value, the jitter becomes dominated by the $1/f^3$ noise component. Lastly, for N greater than a second value of N, a saturation effect is evident, so the jitter does not grow anymore when considering a longer time interval (jitter stable increasing the number of clock periods N). The PLL loop acts as a high pass filter for any noise in the loop, with cutoff frequency equal to the bandwidth of the PLL. The saturation value corresponds to what has been called self referred long term jitter of the PLL. In [Figure 53](#) the maximum jitter trend versus the number of clock periods N (for some typical CPU frequencies) is shown: The curves represent the very worst case, computed taking into account all corners of temperature, power supply and process variations; the real jitter is always measured well below the given worst case values.

Noise in supply and substrate

Digital supply noise adds determining elements to PLL output jitter, independent of the multiplication factor. Its effect is strongly reduced thanks to particular care used in the physical implementation and integration of the PLL module inside the device. In any case, the contribution of digital noise to global jitter is widely taken into account in the curves provided in [Figure 53](#).

Figure 53. ST10F276 PLL jitter



23.8.12 PLL lock/unlock

During normal operation, if the PLL is unlocked for any reason, an interrupt request to the CPU is generated and the reference clock (oscillator) is automatically disconnected from the PLL input: In this way, the PLL goes into free-running mode, providing the system with a backup clock signal (free running frequency F_{free}). This feature allows to recover from a crystal failure occurrence without risking to go into an undefined configuration: The system is provided with a clock allowing the execution of the PLL unlock interrupt routine in a safe mode.

The path between the reference clock and PLL input can be restored only by a hardware reset, or by a bidirectional software or watchdog reset event that forces the $\overline{\text{RSTIN}}$ pin low.

Note: The external RC circuit on $\overline{\text{RSTIN}}$ pin must be the right size in order to extend the duration of the low pulse to grant the PLL to be locked before the level at $\overline{\text{RSTIN}}$ pin is recognized high: Bidirectional reset internally drives $\overline{\text{RSTIN}}$ pin low for just 1024 TCL (definitely not sufficient to get the PLL locked starting from free-running mode).

Conditions: $V_{\text{DD}} = 5\text{V} \pm 10\%$, $T_{\text{A}} = -40 / +125^{\circ}\text{C}$

Table 97. PLL lock/unlock timing

| Symbol | Parameter | Conditions | Value | | Unit |
|-------------------|---|---|------------|--------------|---------------|
| | | | Min. | Max. | |
| T_{PSUP} | PLL Start-up time ⁽¹⁾ | Stable V_{DD} and reference clock | – | 300 | μs |
| T_{LOCK} | PLL Lock-in time | Stable V_{DD} and reference clock, starting from free-running mode | – | 250 | |
| T_{JIT} | Single Period Jitter ⁽¹⁾ (cycle to cycle = 2 TCL) | 6 sigma time period variation (peak to peak) | –500 | +500 | ps |
| F_{free} | PLL free running frequency | Multiplication factors: 3, 4 Multiplication factors: 5, 8, 10, 16 | 250 500 | 2000 4000 | kHz |

1. Not 100% tested, guaranteed by design characterization.

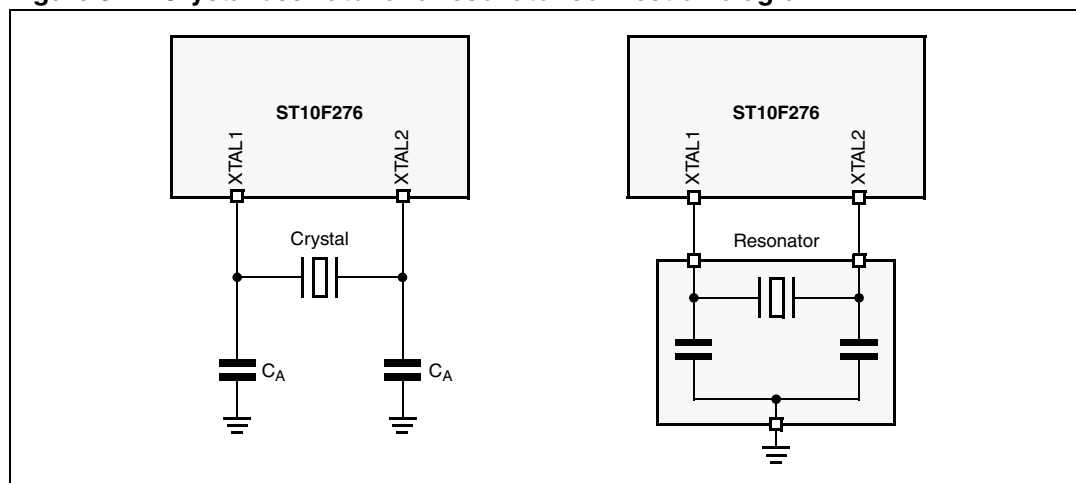
23.8.13 Main oscillator specifications

Conditions: $V_{\text{DD}} = 5\text{V} \pm 10\%$, $T_{\text{A}} = -40 / +125^{\circ}\text{C}$

Table 98. Main oscillator specifications

| Symbol | Parameter | Conditions | Value | | | Unit |
|-------------------|--|------------------------------------|-------|----------------------------|------|------|
| | | | Min. | Typ. | Max. | |
| g_{m} | Oscillator transconductance | | 8 | 17 | 35 | mA/V |
| V_{OSC} | Oscillation amplitude ⁽¹⁾ | Peak to peak | – | $V_{\text{DD}} - 0.4$ | – | V |
| V_{AV} | Oscillation voltage level ⁽¹⁾ | Sine wave middle | – | $V_{\text{DD}} / 2 - 0.25$ | – | |
| t_{STUP} | Oscillator start-up time ⁽¹⁾ | Stable V_{DD} - crystal | – | 3 | 4 | ms |
| | | Stable V_{DD} , resonator | – | 2 | 3 | |

1. Not 100% tested, guaranteed by design characterization

Figure 54. Crystal oscillator and resonator connection diagram**Table 99. Negative resistance (absolute min. value @125°C / $V_{DD} = 4.5V$)**

| C_A (pF) | 12 | 15 | 18 | 22 | 27 | 33 | 39 | 47 |
|------------|--------------|--------------|--------------|--------------|--------------|---------------|---------------|---------------|
| 4 MHz | 460 Ω | 550 Ω | 675 Ω | 800 Ω | 840 Ω | 1000 Ω | 1180 Ω | 1200 Ω |
| 8 MHz | 380 Ω | 460 Ω | 540 Ω | 640 Ω | 580 Ω | - | - | - |
| 12 MHz | 370 Ω | 420 Ω | 360 Ω | - | - | - | - | - |

The given values of C_A do not include the stray capacitance of the package and of the printed circuit board: The negative resistance values are calculated assuming additional 5pF to the values in the table. The crystal shunt capacitance (C_0), the package and the stray capacitance between XTAL1 and XTAL2 pins is globally assumed equal to 4pF.

The external resistance between XTAL1 and XTAL2 is not necessary, since already present on the silicon.

23.8.14 32 kHz Oscillator specifications

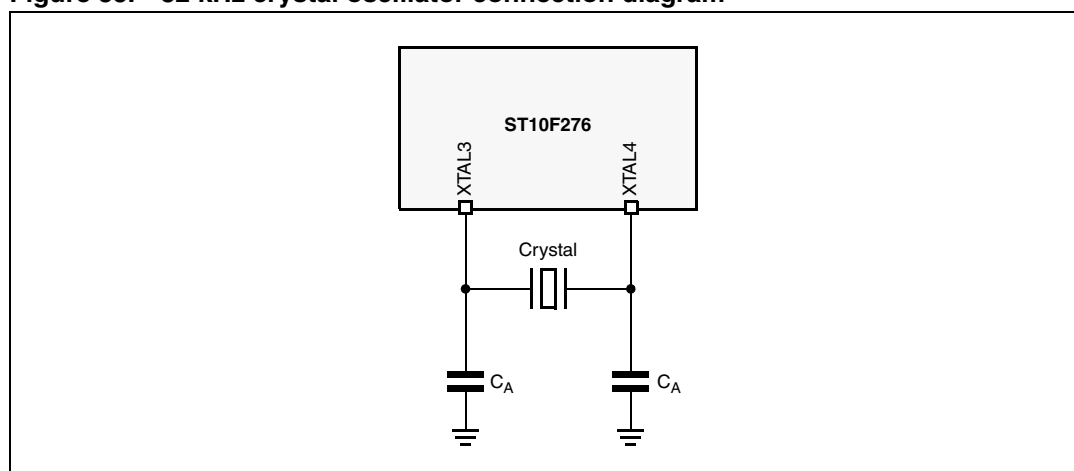
Conditions: $V_{DD} = 5V \pm 10\%$, $T_A = -40 / +125^\circ C$

Table 100. 32 kHz Oscillator specifications

| Symbol | Parameter | Conditions | Value | | | Unit |
|--------------|--|------------------|-------|------|------|-----------|
| | | | Min. | Typ. | Max. | |
| g_{m32} | Oscillator ⁽¹⁾ | Start-up | 20 | 31 | 50 | $\mu A/V$ |
| | | Normal run | 8 | 17 | 30 | |
| V_{OSC32} | Oscillation amplitude ⁽²⁾ | Peak to peak | 0.5 | 1.0 | 2.4 | V |
| V_{AV32} | Oscillation voltage level ⁽²⁾ | Sine wave middle | 0.7 | 0.9 | 1.2 | |
| t_{STUP32} | Oscillator start-up time ⁽²⁾ | Stable V_{DD} | — | 1 | 5 | s |

1. At power-on a high current biasing is applied for faster oscillation start-up. Once the oscillation is started, the current biasing is reduced to lower the power consumption of the system.

2. Not 100% tested, guaranteed by design characterization.

Figure 55. 32 kHz crystal oscillator connection diagram**Table 101. Minimum values of negative resistance (module)**

| | $C_A = 6\text{pF}$ | $C_A = 12\text{pF}$ | $C_A = 15\text{pF}$ | $C_A = 18\text{pF}$ | $C_A = 22\text{pF}$ | $C_A = 27\text{pF}$ | $C_A = 33\text{pF}$ |
|--------|--------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| 32 kHz | - | - | - | - | 150 k Ω | 120 k Ω | 90 k Ω |

The given values of C_A do not include the stray capacitance of the package and of the printed circuit board: The negative resistance values are calculated assuming additional 5pF to the values in the table. The crystal shunt capacitance (C_0), the package and the stray capacitance between XTAL3 and XTAL4 pins is globally assumed equal to 4pF. The external resistance between XTAL3 and XTAL4 is not necessary, since already present on the silicon.

Warning: Direct driving on XTAL3 pin is not supported. Always use a 32 kHz crystal oscillator.

23.8.15 External clock drive XTAL1

When Direct Drive configuration is selected during reset, it is possible to drive the CPU clock directly from the XTAL1 pin, without particular restrictions on the maximum frequency, since the on-chip oscillator amplifier is bypassed. The speed limit is imposed by internal logic that targets a maximum CPU frequency of 64 MHz.

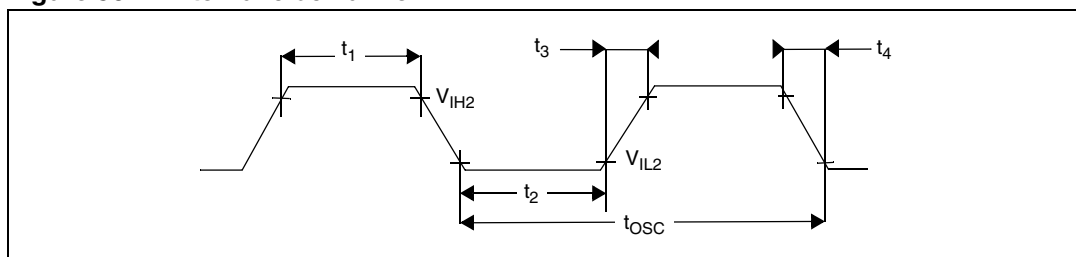
In all other clock configurations (Direct Drive with Prescaler or PLL usage) the on-chip oscillator amplifier is not bypassed, so it determines the input clock speed limit. Then an external clock source can be used but limited in the range of frequencies defined for the usage of crystal and resonator (refer also to [Table 95 on page 196](#)).

External clock drive timing conditions: $V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$

Table 102. External clock drive timing

| Symbol | Parameter | Direct drive $f_{CPU} = f_{XTAL}$ | | Direct drive with prescaler $f_{CPU} = f_{XTAL} / 2$ | | PLL usage $f_{CPU} = f_{XTAL} \times F$ | | Unit |
|---------------------|---------------------------------|--------------------------------------|------|--|------|--|------|------|
| | | Min. | Max. | Min. | Max. | Min. | Max. | |
| t_{OSC} SR | XTAL1 period ^{(1) (2)} | 15.625 | — | 83.3 | 250 | 83.3 | 250 | ns |
| t_1 SR | High time ⁽³⁾ | 6 | — | 3 | — | 6 | — | |
| t_2 SR | Low time ⁽³⁾ | | | | | | | |
| t_3 SR | Rise time ⁽³⁾ | — | 2 | — | 2 | — | 2 | |
| t_4 SR | Fall time ⁽³⁾ | | | | | | | |

1. The minimum value for the XTAL1 signal period is considered as the theoretical minimum. The real minimum value depends on the duty cycle of the input clock signal.
2. 4-12 MHz is the input frequency range when using an external clock source. 64 MHz can be applied with an external clock source only when Direct Drive mode is selected: In this case, the oscillator amplifier is bypassed so it does not limit the input frequency.
3. The input clock signal must reach the defined levels V_{IL2} and V_{IH2} .

Figure 56. External clock drive XTAL1

Note: When Direct Drive is selected, an external clock source can be used to drive XTAL1. The maximum frequency of the external clock source depends on the duty cycle: When 64 MHz is used, 50% duty cycle is granted (low phase = high phase = 7.8ns); when for instance 32 MHz is used, a 25% duty cycle can be accepted (minimum phase, high or low, again equal to 7.8ns).

23.8.16 Memory cycle variables

The tables below use three variables which are derived from the BUSCONx registers and represent the special characteristics of the programmed memory cycle. The following table describes how these variables are computed.

Table 103. Memory cycle variables

| Symbol | Description | Values |
|--------|-------------------------------|-----------------------------|
| t_A | ALE extension | $TCL \times [ALECTL]$ |
| t_C | Memory cycle time wait states | $2TCL \times (15 - [MCTC])$ |
| t_F | Memory tri-state time | $2TCL \times (1 - [MTTC])$ |

23.8.17 External memory bus timing

In the next sections the External Memory Bus timings are described. The given values are computed for a maximum CPU clock of 40 MHz.

It is evident that when higher CPU clock frequency is used (up to 64 MHz), some numbers in the timing formulas become zero or negative, which in most cases is not acceptable or meaningful. In these cases, the speed of the bus settings t_A , t_C and t_F must be correctly adjusted.

Note: All External Memory Bus Timings and SSC Timings presented in the following tables are given by design characterization and not fully tested in production.

23.8.18 Multiplexed bus

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$, $C_L = 50\text{pF}$,

ALE cycle time = $6 \text{ TCL} + 2t_A + t_C + t_F$ (75ns at 40 MHz CPU clock without wait states).

Table 104. Multiplexed bus

| Symbol | Parameter | F _{CPU} = 40 MHz TCL = 12.5ns | | Variable CPU clock 1/2 TCL = 1 to 64 MHz | | Unit |
|--------------------|--|---|---|---|---|------|
| | | Min. | Max. | Min. | Max. | |
| t ₅ CC | ALE high time | 4 + t _A | – | TCL – 8.5 + t _A | – | ns |
| t ₆ CC | Address setup to ALE | 1.5 + t _A | | TCL – 11 + t _A | | |
| t ₇ CC | Address hold after ALE | 4 + t _A | | TCL – 8.5 + t _A | | |
| t ₈ CC | ALE falling edge to $\overline{\text{RD}}$, $\overline{\text{WR}}$ (with RW-delay) | 4 + t _A | | TCL – 8.5 + t _A | | |
| t ₉ CC | ALE falling edge to $\overline{\text{RD}}$, $\overline{\text{WR}}$ (no RW-delay) | – 8.5 + t _A | | – 8.5 + t _A | | |
| t ₁₀ CC | Address float after $\overline{\text{RD}}$, $\overline{\text{WR}}$ (with RW-delay) ⁽¹⁾ | – | 6 | – | 6 | |
| t ₁₁ CC | Address float after $\overline{\text{RD}}$, $\overline{\text{WR}}$ (no RW-delay) ¹ | | 18.5 | | TCL + 6 | |
| t ₁₂ CC | $\overline{\text{RD}}$, $\overline{\text{WR}}$ low time (with RW-delay) | 15.5 + t _C | – | 2TCL – 9.5 + t _C | – | |
| t ₁₃ CC | $\overline{\text{RD}}$, $\overline{\text{WR}}$ low time (no RW-delay) | 28 + t _C | | 3TCL – 9.5 + t _C | | |
| t ₁₄ SR | $\overline{\text{RD}}$ to valid data in (with RW-delay) | – | 6 + t _C | – | 2TCL – 19 + t _C | |
| t ₁₅ SR | $\overline{\text{RD}}$ to valid data in (no RW-delay) | | 18.5 + t _C | | 3TCL – 19 + t _C | |
| t ₁₆ SR | ALE low to valid data in | | 17.5 + + t _A + t _C | | 3TCL – 20 + + t _A + t _C | |
| t ₁₇ SR | Address/Unlatched $\overline{\text{CS}}$ to valid data in | | 20 + 2t _A + + t _C | | 4TCL – 30 + + 2t _A + t _C | |
| t ₁₈ SR | Data hold after $\overline{\text{RD}}$ rising edge | 0 | – | 0 | – | |
| t ₁₉ SR | Data float after $\overline{\text{RD}}$ ¹ | – | 16.5 + t _F | – | 2TCL – 8.5 + t _F | |
| t ₂₂ CC | Data valid to $\overline{\text{WR}}$ | 10 + t _C | – | 2TCL – 15 + t _C | – | |
| t ₂₃ CC | Data hold after $\overline{\text{WR}}$ | 4 + t _F | | 2TCL – 8.5 + t _F | | |
| t ₂₅ CC | ALE rising edge after $\overline{\text{RD}}$, $\overline{\text{WR}}$ | 15 + t _F | | 2TCL – 10 + t _F | | |
| t ₂₇ CC | Address/Unlatched $\overline{\text{CS}}$ hold after $\overline{\text{RD}}$, $\overline{\text{WR}}$ | 10 + t _F | | 2TCL – 15 + t _F | | |

Table 104. Multiplexed bus (continued)

| Symbol | Parameter | F _{CPU} = 40 MHz TCL = 12.5ns | | Variable CPU clock 1/2 TCL = 1 to 64 MHz | | Unit |
|---------------------------|---|---|---|---|--|------|
| | | Min. | Max. | Min. | Max. | |
| t ₃₈ CC | ALE falling edge to Latched CS | − 4 − t _A | 10 − t _A | − 4 − t _A | 10 − t _A | ns |
| t ₃₉ SR | Latched CS low to valid data In | − | 16.5 + t _C + 2t _A | − | 3TCL− 21+ t _C + 2t _A | |
| t ₄₀ CC | Latched CS hold after RD, WR | 27 + t _F | − | 3TCL − 10.5 + t _F | − | |
| t ₄₂ CC | ALE fall. edge to RdCS, WrCS (with RW delay) | 7 + t _A | | TCL − 5.5 + t _A | | |
| t ₄₃ CC | ALE fall. edge to RdCS, WrCS (no RW delay) | − 5.5 + t _A | | − 5.5 + t _A | | |
| t ₄₄ CC | Address float after RdCS, WrCS (with RW delay) ¹ | − | 1.5 | − | 1.5 | |
| t ₄₅ CC | Address float after RdCS, WrCS (no RW delay) | | 14 | | TCL + 1.5 | |
| t ₄₆ SR | RdCS to valid data In (with RW delay) | | 4 + t _C | | 2TCL − 21 + t _C | |
| t ₄₇ SR | RdCS to valid data In (no RW delay) | | 16.5 + t _C | | 3TCL − 21 + t _C | |
| t ₄₈ CC | RdCS, WrCS low time (with RW delay) | 15.5 + t _C | − | 2TCL − 9.5 + t _C | − | |
| t ₄₉ CC | RdCS, WrCS low time (no RW delay) | 28 + t _C | | 3TCL − 9.5 + t _C | | |
| t ₅₀ CC | Data valid to WrCS | 10 + t _C | | 2TCL − 15 + t _C | | |
| t ₅₁ SR | Data hold after RdCS | 0 | | 0 | | |
| t ₅₂ SR | Data float after RdCS ⁽¹⁾ | − | 16.5 + t _F | − | 2TCL − 8.5 + t _F | |
| t ₅₄ CC | Address hold after RdCS, WrCS | 6 + t _F | − | 2TCL − 19 + t _F | − | |
| t ₅₆ CC | Data hold after WrCS | | | | | |

1. Partially tested, guaranteed by design characterization.

Figure 57. Multiplexed bus with/without R/W delay and normal ALE

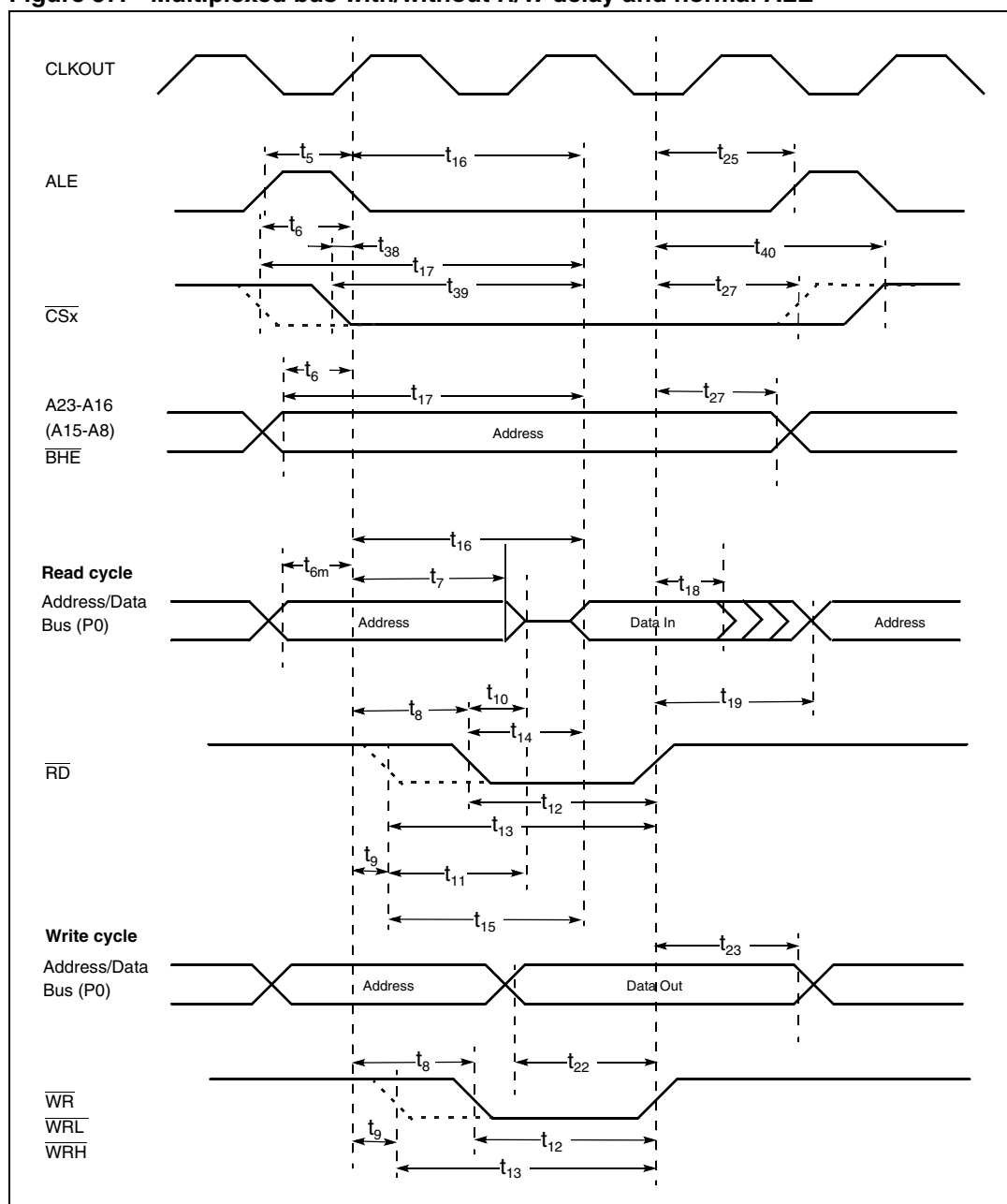


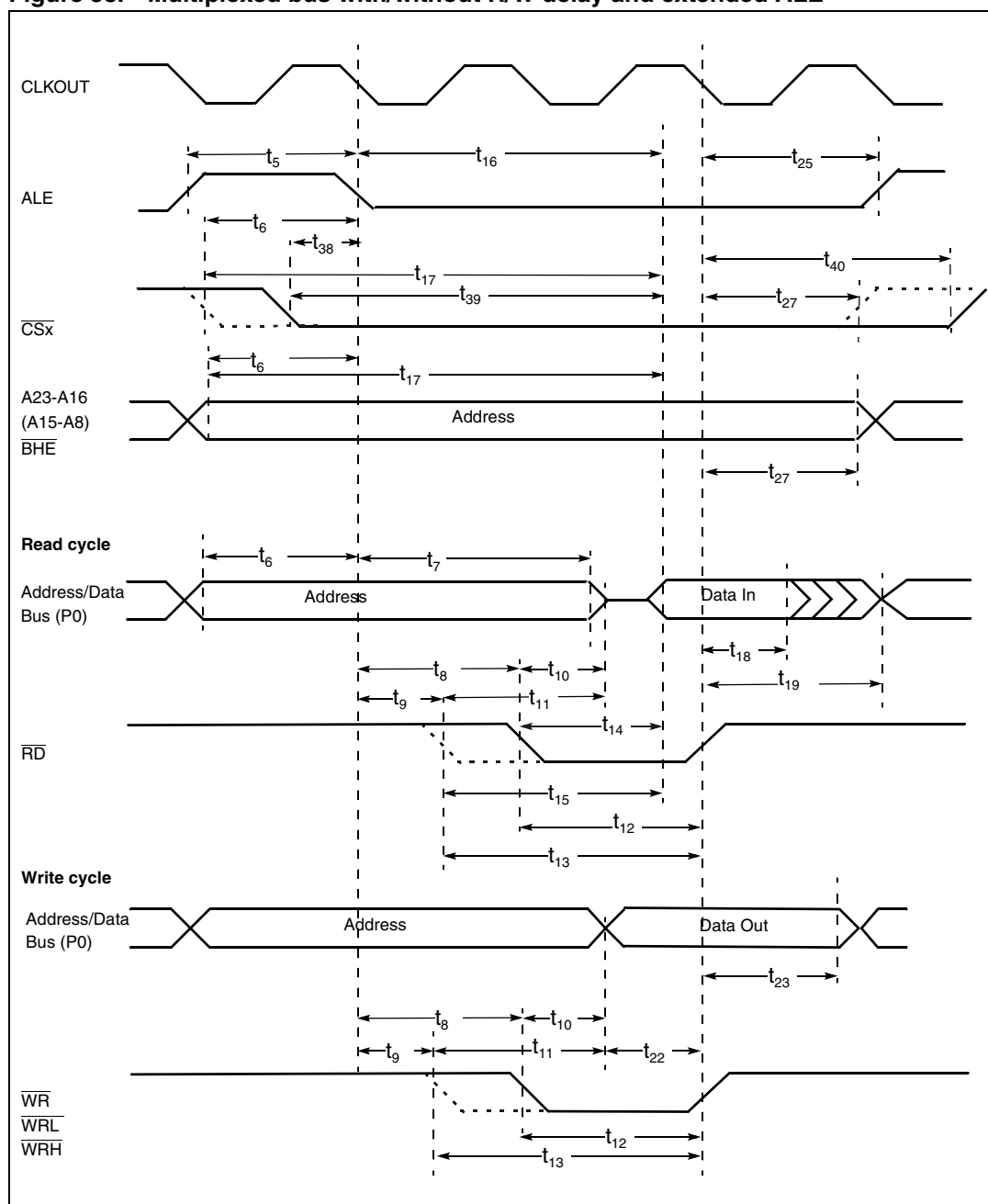
Figure 58. Multiplexed bus with/without R/W delay and extended ALE

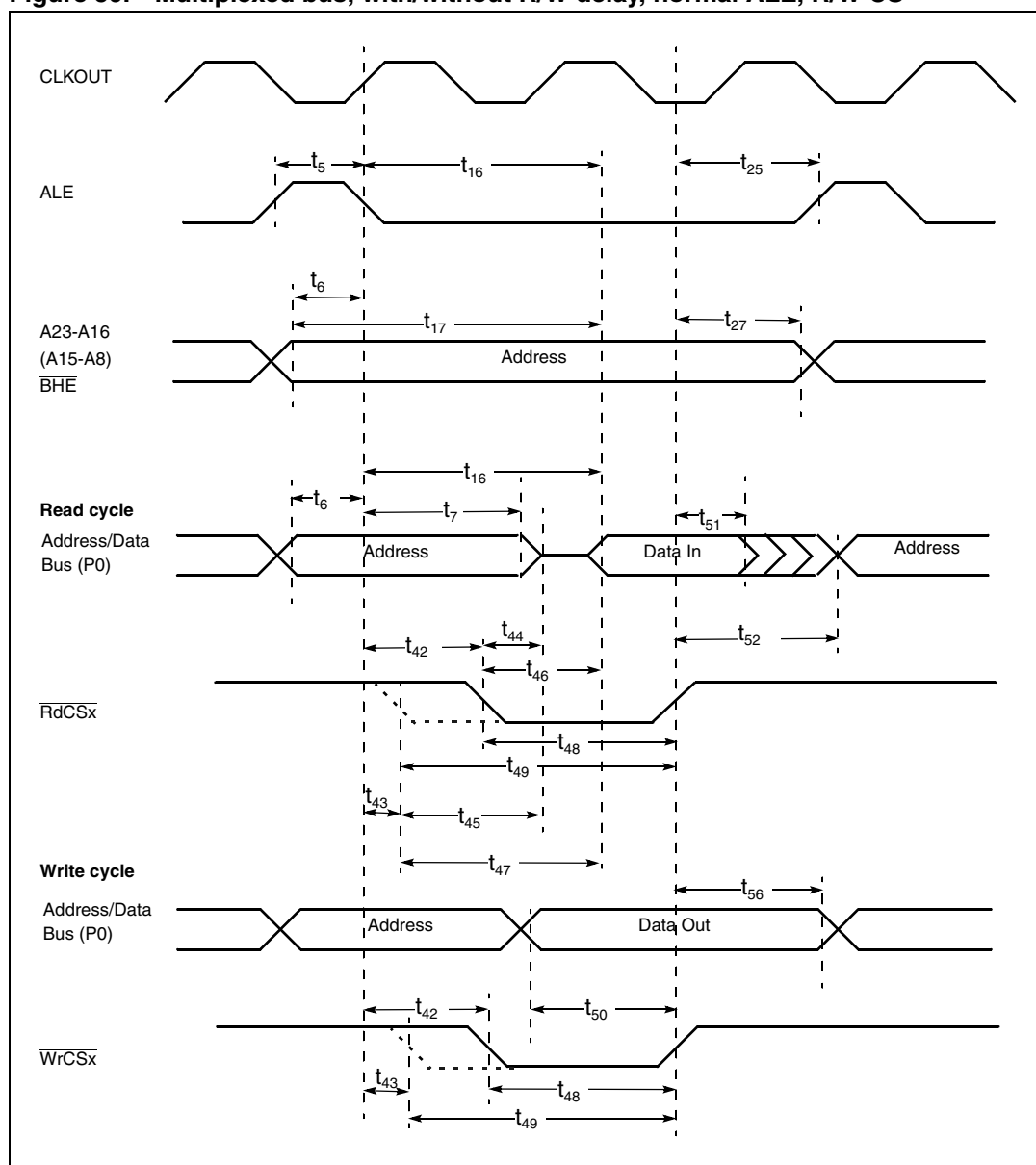
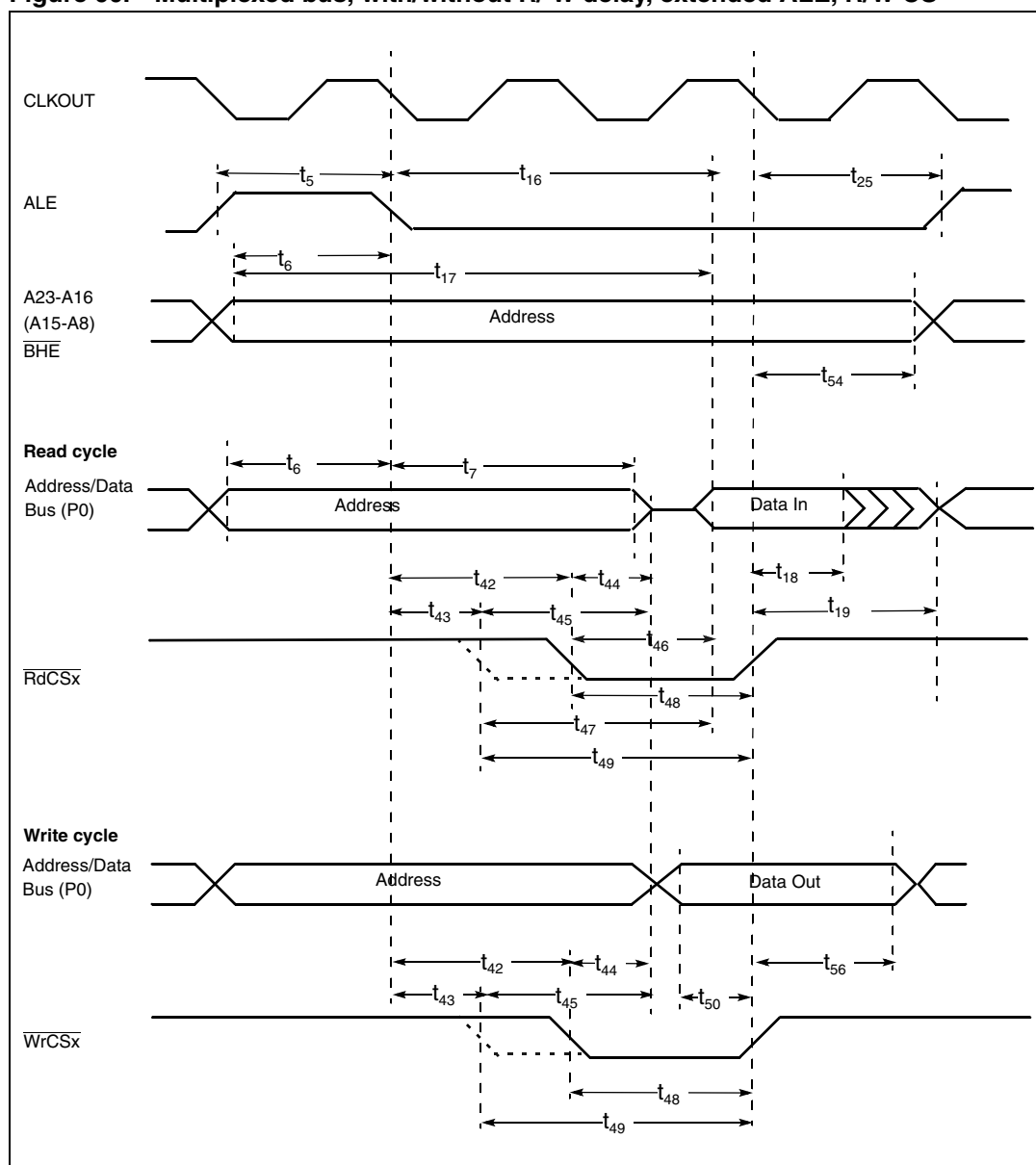
Figure 59. Multiplexed bus, with/without R/W delay, normal ALE, R/W CS

Figure 60. Multiplexed bus, with/without R/ W delay, extended ALE, R/W CS

23.8.19 Demultiplexed bus

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ C$, $C_L = 50pF$,

ALE cycle time = $4 \text{ TCL} + 2t_A + t_C + t_F$ (50ns at 40 MHz CPU clock without wait states).

Table 105. Demultiplexed bus

| Symbol | Parameter | $F_{CPU} = 40 \text{ MHz}$ $TCL = 12.5ns$ | | Variable CPU clock $1/2 \text{ TCL} = 1 \text{ to } 64 \text{ MHz}$ | | Unit |
|--------------|---|--|--------------------|--|---------------------------|------|
| | | Min. | Max. | Min. | Max. | |
| t_5 CC | ALE high time | $4 + t_A$ | – | $TCL - 8.5 + t_A$ | – | ns |
| t_6 CC | Address setup to ALE | $1.5 + t_A$ | – | $TCL - 11 + t_A$ | – | ns |
| t_{80} CC | Address/Unlatched \overline{CS} setup to \overline{RD} , \overline{WR} (with RW-delay) | $12.5 + 2t_A$ | – | $2TCL - 12.5 + 2t_A$ | – | ns |
| t_{81} CC | Address/Unlatched \overline{CS} setup to \overline{RD} , \overline{WR} (no RW-delay) | $0.5 + 2t_A$ | – | $TCL - 12 + 2t_A$ | – | ns |
| t_{12} CC | \overline{RD} , \overline{WR} low time (with RW-delay) | $15.5 + t_C$ | – | $2TCL - 9.5 + t_C$ | – | ns |
| t_{13} CC | \overline{RD} , \overline{WR} low time (no RW-delay) | $28 + t_C$ | – | $3TCL - 9.5 + t_C$ | – | ns |
| t_{14} SR | \overline{RD} to valid data in (with RW-delay) | – | $6 + t_C$ | – | $2TCL - 19 + t_C$ | ns |
| t_{15} SR | \overline{RD} to valid data in (no RW-delay) | – | $18.5 + t_C$ | – | $3TCL - 19 + t_C$ | ns |
| t_{16} SR | ALE low to valid data in | – | $17.5 + t_A + t_C$ | – | $3TCL - 20 + t_A + t_C$ | ns |
| t_{17} SR | Address/Unlatched \overline{CS} to valid data in | – | $20 + 2t_A + t_C$ | – | $4TCL - 30 + 2t_A + t_C$ | ns |
| t_{18} SR | Data hold after \overline{RD} rising edge | 0 | – | 0 | – | ns |
| t_{20} SR | Data float after \overline{RD} rising edge (with RW-delay) ⁽¹⁾ | – | $16.5 + t_F$ | – | $2TCL - 8.5 + t_F + 2t_A$ | ns |
| t_{21} SR | Data float after \overline{RD} rising edge (no RW-delay) ¹ | – | $4 + t_F$ | – | $TCL - 8.5 + t_F + 2t_A$ | ns |
| t_{22} CC | Data valid to \overline{WR} | $10 + t_C$ | – | $2TCL - 15 + t_C$ | – | ns |
| t_{24} CC | Data hold after \overline{WR} | $4 + t_F$ | – | $TCL - 8.5 + t_F$ | – | ns |
| t_{26} CC | ALE rising edge after \overline{RD} , \overline{WR} | $-10 + t_F$ | – | $-10 + t_F$ | – | ns |
| t_{28} CC | Address/Unlatched \overline{CS} hold after \overline{RD} , \overline{WR} ⁽²⁾ | $0 + t_F$ | – | $0 + t_F$ | – | ns |
| t_{28h} CC | Address/Unlatched \overline{CS} hold after \overline{WRH} | $-5 + t_F$ | – | $-5 + t_F$ | – | ns |

Table 105. Demultiplexed bus (continued)

| Symbol | Parameter | F _{CPU} = 40 MHz TCL = 12.5ns | | Variable CPU clock 1/2 TCL = 1 to 64 MHz | | Unit |
|--------------------|---|---|---------------------|---|--------------------------|------|
| | | Min. | Max. | Min. | Max. | |
| t ₃₈ CC | ALE falling edge to Latched $\overline{\text{CS}}$ | $-4 - t_A$ | $6 - t_A$ | $-4 - t_A$ | $6 - t_A$ | ns |
| t ₃₉ SR | Latched $\overline{\text{CS}}$ low to Valid Data In | – | $16.5 + t_C + 2t_A$ | – | $3TCL - 21 + t_C + 2t_A$ | ns |
| t ₄₁ CC | Latched $\overline{\text{CS}}$ hold after $\overline{\text{RD}}$, $\overline{\text{WR}}$ | $2 + t_F$ | – | $TCL - 10.5 + t_F$ | – | ns |
| t ₈₂ CC | Address setup to $\overline{\text{RdCS}}$, $\overline{\text{WrCS}}$ (with RW-delay) | $14 + 2t_A$ | – | $2TCL - 11 + 2t_A$ | – | ns |
| t ₈₃ CC | Address setup to $\overline{\text{RdCS}}$, $\overline{\text{WrCS}}$ (no RW-delay) | $2 + 2t_A$ | – | $TCL - 10.5 + 2t_A$ | – | ns |
| t ₄₆ SR | $\overline{\text{RdCS}}$ to Valid Data In (with RW-delay) | – | $4 + t_C$ | – | $2TCL - 21 + t_C$ | ns |
| t ₄₇ SR | $\overline{\text{RdCS}}$ to Valid Data In (no RW-delay) | – | $16.5 + t_C$ | – | $3TCL - 21 + t_C$ | ns |
| t ₄₈ CC | $\overline{\text{RdCS}}$, $\overline{\text{WrCS}}$ low time (with RW-delay) | $15.5 + t_C$ | – | $2TCL - 9.5 + t_C$ | – | ns |
| t ₄₉ CC | $\overline{\text{RdCS}}$, $\overline{\text{WrCS}}$ low time (no RW-delay) | $28 + t_C$ | – | $3TCL - 9.5 + t_C$ | – | ns |
| t ₅₀ CC | Data valid to $\overline{\text{WrCS}}$ | $10 + t_C$ | – | $2TCL - 15 + t_C$ | – | ns |
| t ₅₁ SR | Data hold after $\overline{\text{RdCS}}$ | 0 | – | 0 | – | ns |
| t ₅₃ SR | Data float after $\overline{\text{RdCS}}$ (with RW-delay) | – | $16.5 + t_F$ | – | $2TCL - 8.5 + t_F$ | ns |
| t ₆₈ SR | Data float after $\overline{\text{RdCS}}$ (no RW-delay) | – | $4 + t_F$ | – | $TCL - 8.5 + t_F$ | ns |
| t ₅₅ CC | Address hold after $\overline{\text{RdCS}}$, $\overline{\text{WrCS}}$ | $-8.5 + t_F$ | – | $-8.5 + t_F$ | – | ns |
| t ₅₇ CC | Data hold after $\overline{\text{WrCS}}$ | $2 + t_F$ | – | $TCL - 10.5 + t_F$ | – | ns |

1. RW-delay and t_A refer to the next following bus cycle.
2. Read data is latched with the same clock edge that triggers the address change and the rising $\overline{\text{RD}}$ edge. Therefore address changes which occur before the end of $\overline{\text{RD}}$ have no impact on read cycles.

The following figures ([Figure 57](#) to [Figure 64](#)) present the different configurations of external memory cycle.

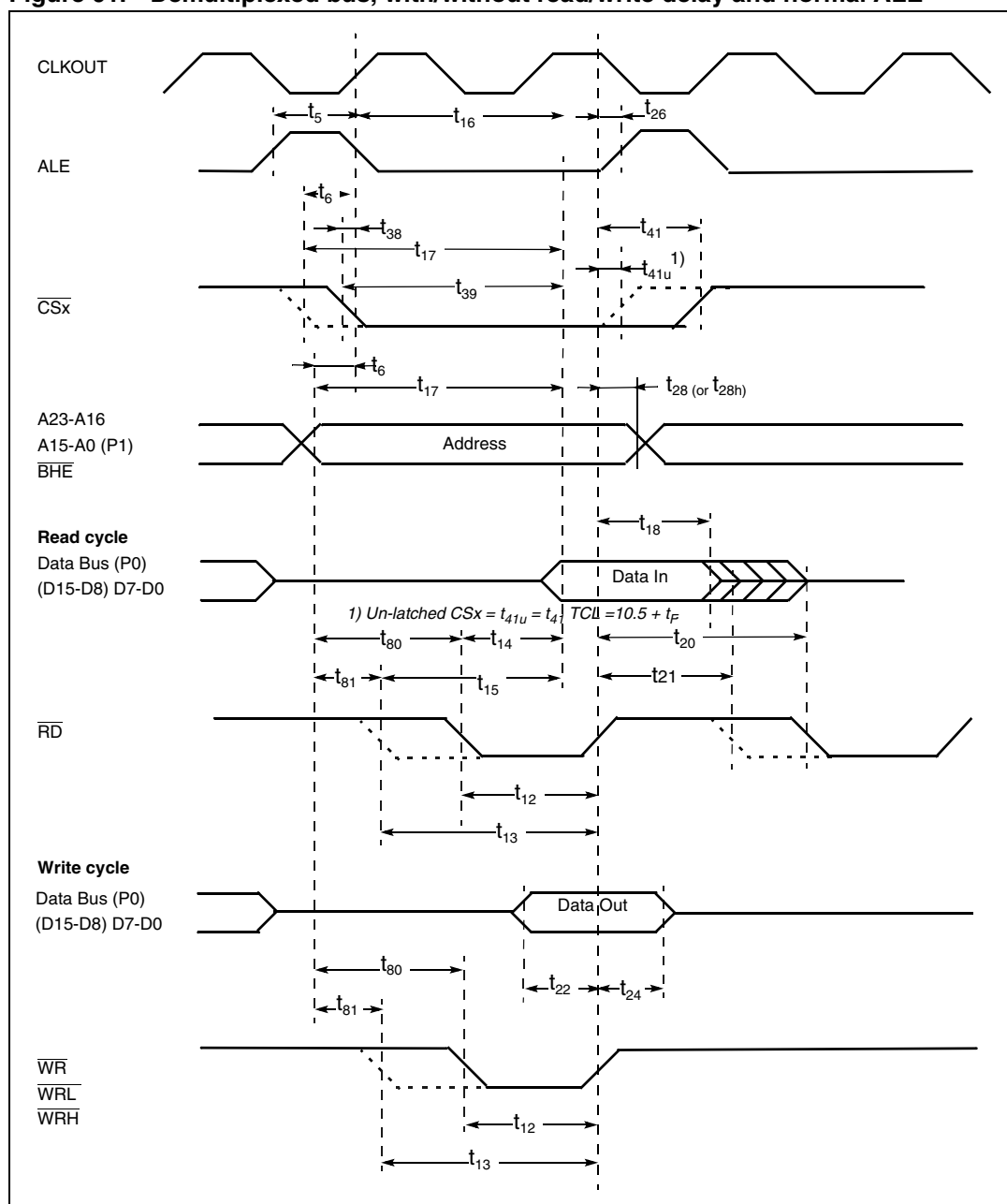
Figure 61. Demultiplexed bus, with/without read/write delay and normal ALE

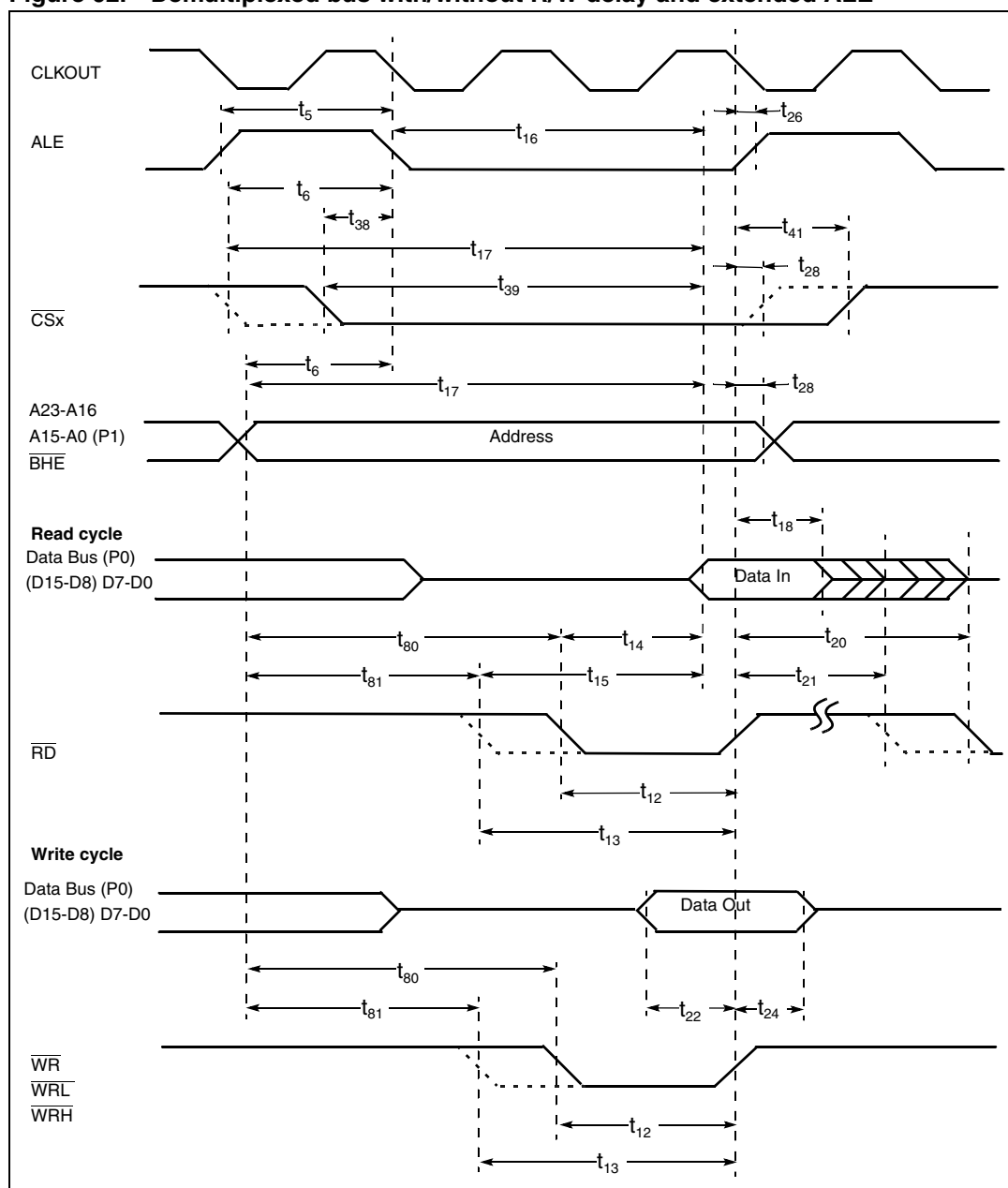
Figure 62. Demultiplexed bus with/without R/W delay and extended ALE

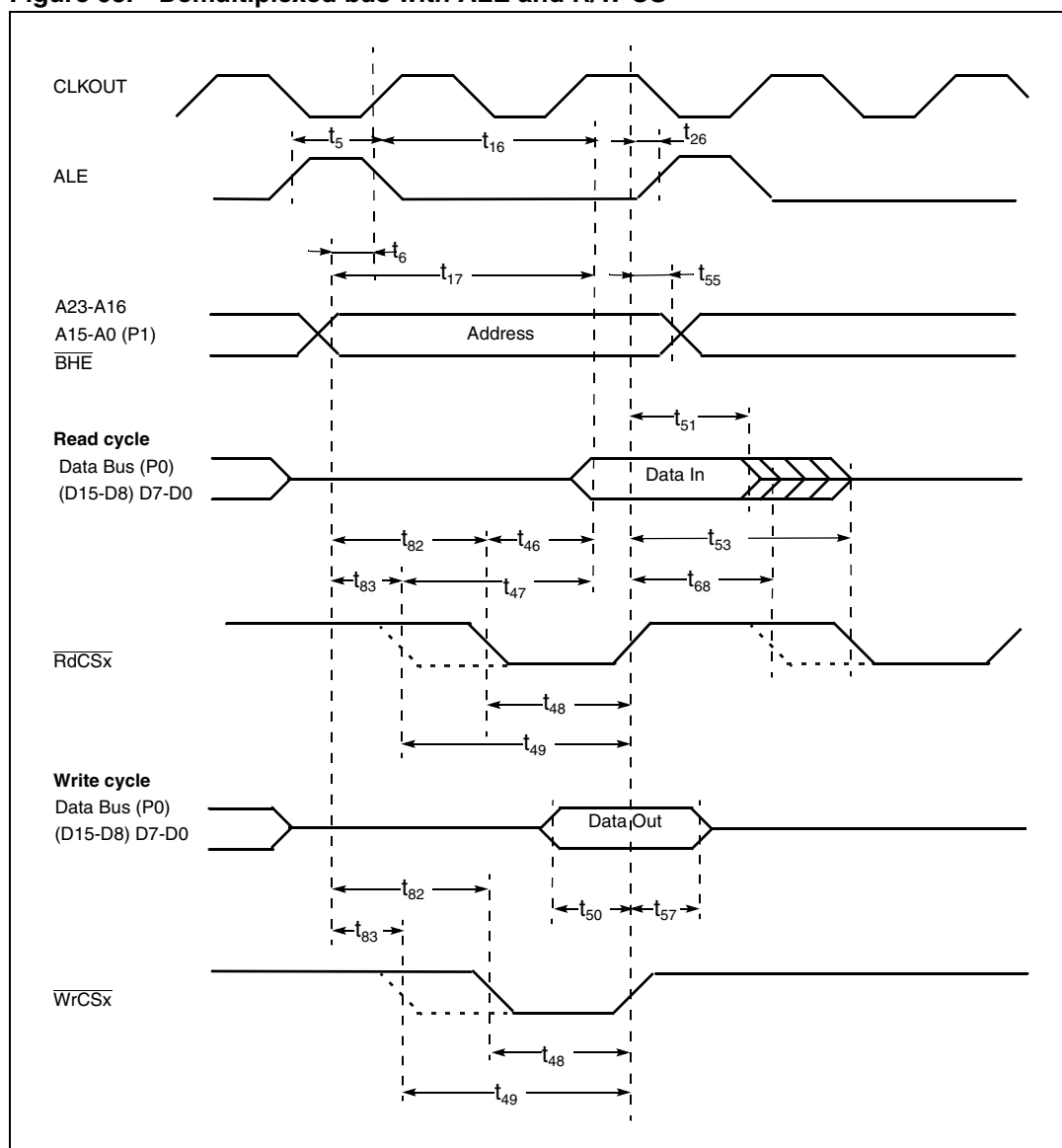
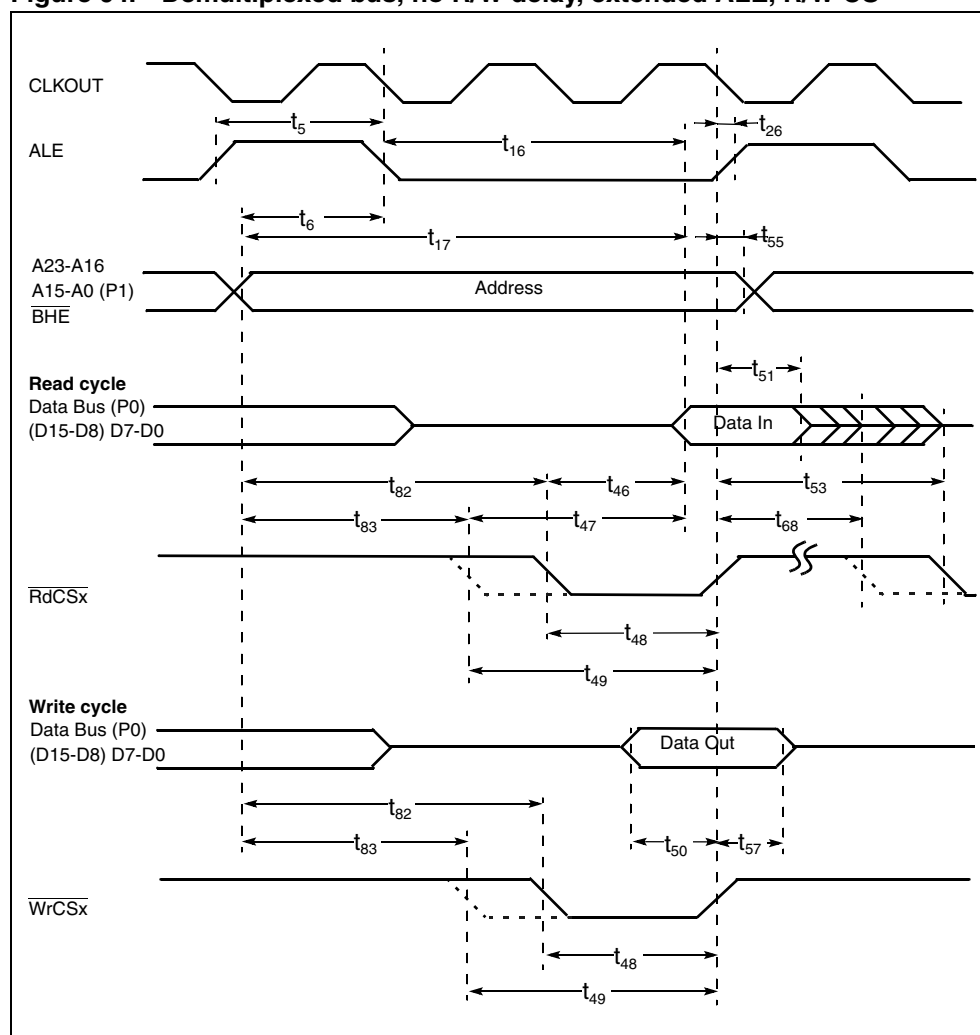
Figure 63. Demultiplexed bus with ALE and R/W CS

Figure 64. Demultiplexed bus, no R/W delay, extended ALE, R/W CS

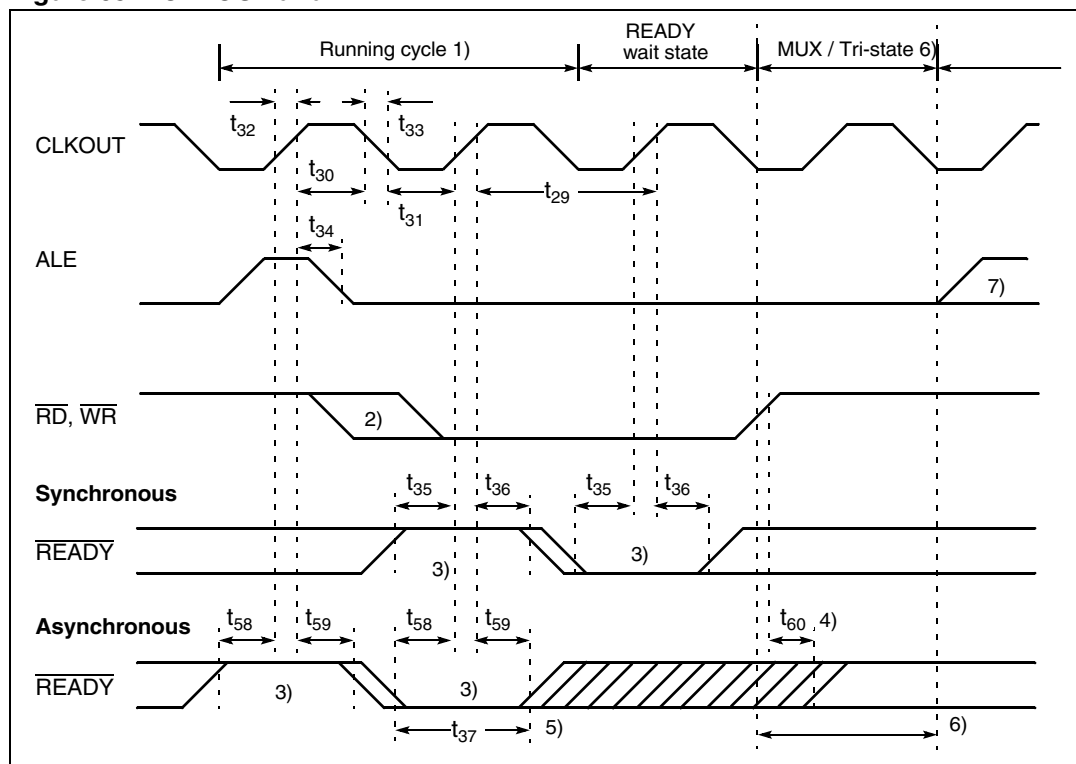
23.8.20 CLKOUT and $\overline{\text{READY}}$

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$, $C_L = 50\text{pF}$

Table 106. CLKOUT and $\overline{\text{READY}}$

| Symbol | Parameter | F _{CPU} = 40 MHz TCL = 12.5ns | | Variable CPU clock 1/2 TCL = 1 to 64 MHz | | Unit |
|---------------------------|---|---|---|---|---|------|
| | | Min. | Max. | Min. | Max. | |
| t ₂₉ CC | CLKOUT cycle time | 25 | 25 | 2TCL | 2TCL | ns |
| t ₃₀ CC | CLKOUT high time | 9 | – | TCL – 3.5 | – | |
| t ₃₁ CC | CLKOUT low time | 10 | | TCL – 2.5 | | |
| t ₃₂ CC | CLKOUT rise time | – | 4 | – | 4 | |
| t ₃₃ CC | CLKOUT fall time | | | | | |
| t ₃₄ CC | CLKOUT rising edge to ALE falling edge | – 2 + t _A | 8 + t _A | – 2 + t _A | 8 + t _A | |
| t ₃₅ SR | Synchronous $\overline{\text{READY}}$ setup time to CLKOUT | 17 | – | 17 | – | |
| t ₃₆ SR | Synchronous $\overline{\text{READY}}$ hold time after CLKOUT | 2 | | 2 | | |
| t ₃₇ SR | Asynchronous $\overline{\text{READY}}$ low time | 35 | | 2TCL + 10 | | |
| t ₅₈ SR | Asynchronous $\overline{\text{READY}}$ setup time ⁽¹⁾ | 17 | | 17 | | |
| t ₅₉ SR | Asynchronous $\overline{\text{READY}}$ hold time ⁽¹⁾ | 2 | | 2 | | |
| t ₆₀ SR | Async. $\overline{\text{READY}}$ hold time after $\overline{\text{RD}}$, $\overline{\text{WR}}$ high (Demultiplexed Bus) ⁽²⁾ | 0 | 2t _A + t _C + t _F | 0 | 2t _A + t _C + t _F | |

1. These timings are given for characterization purposes only, in order to assure recognition at a specific clock edge.
2. Demultiplexed bus is the worst case. For multiplexed bus 2TCLs must be added to the maximum values. This adds even more time for deactivating $\overline{\text{READY}}$. $2t_A$ and t_C refer to the next following bus cycle and t_F refers to the current bus cycle.

Figure 65. CLKOUT and $\overline{\text{READY}}$ 

1. Cycle as programmed, including MCTC wait states (Example shows 0 MCTC WS).
2. The leading edge of the respective command depends on RW-delay.
3. $\overline{\text{READY}}$ sampled HIGH at this sampling point generates a READY controlled wait state, $\overline{\text{READY}}$ sampled LOW at this sampling point terminates the currently running bus cycle.
4. $\overline{\text{READY}}$ may be deactivated in response to the trailing (rising) edge of the corresponding command ($\overline{\text{RD}}$ or $\overline{\text{WR}}$).
5. If the Asynchronous $\overline{\text{READY}}$ signal does not fulfill the indicated setup and hold times with respect to CLKOUT (for example, because CLKOUT is not enabled), it must fulfill t_{37} in order to be safely synchronized. This is guaranteed if READY is removed in response to the command (see Note 4).
6. Multiplexed bus modes have a MUX wait state added after a bus cycle, and an additional MTTC wait state may be inserted here.
For a multiplexed bus with MTTC wait state this delay is 2 CLKOUT cycles; for a demultiplexed bus without MTTC wait state this delay is zero.
7. The next external bus cycle may start here.

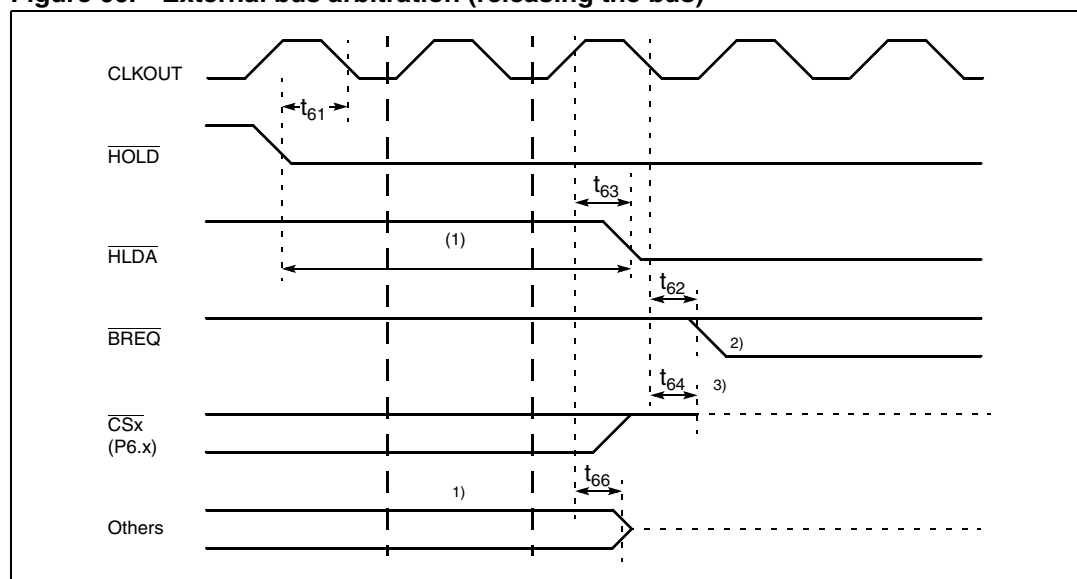
23.8.21 External bus arbitration

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$, $C_L = 50\text{pF}$

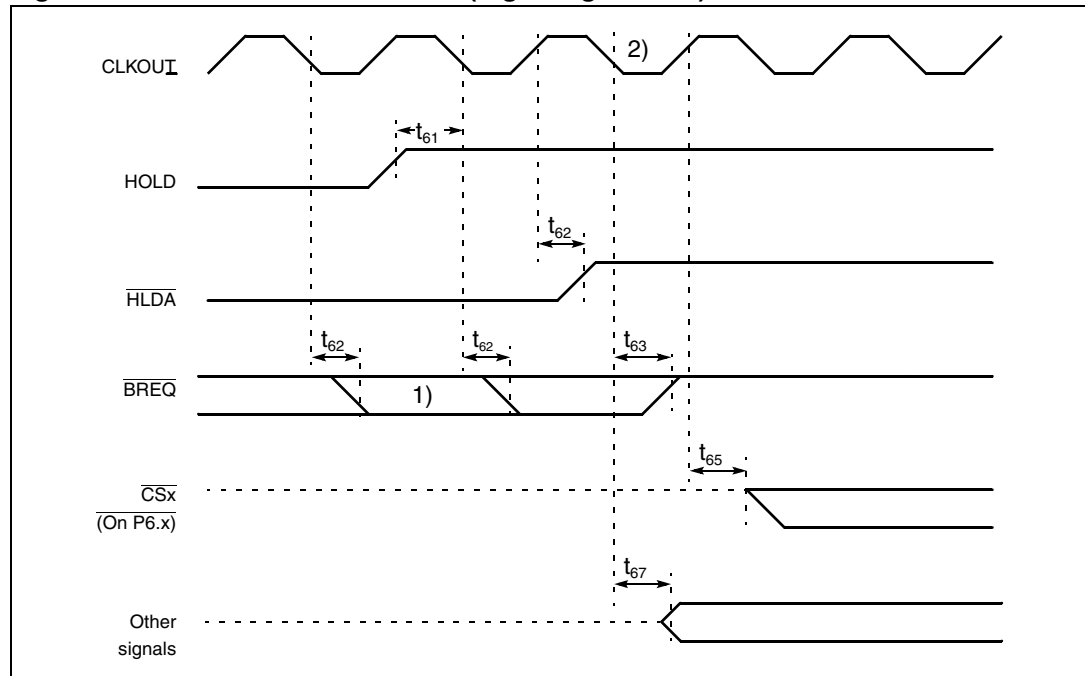
Table 107. External bus arbitration

| Symbol | Parameter | F _{CPU} = 40 MHz TCL = 12.5ns | | Variable CPU Clock 1/2 TCL = 1 to 64 MHz | | Unit |
|---------------------------|---|---|------|---|------|------|
| | | Min. | Max. | Min. | Max. | |
| t ₆₁ SR | HOLD input setup time to CLKOUT | 18.5 | – | 18.5 | – | ns |
| t ₆₂ CC | CLKOUT to H _{LDA} high or B _{REQ} low delay | – | 12.5 | – | 12.5 | |
| t ₆₃ CC | CLKOUT to H _{LDA} low or B _{REQ} high delay | | | | | |
| t ₆₄ CC | C _{Sx} release | | 20 | | 20 | |
| t ₆₅ CC | C _{Sx} drive | – 4 | 15 | – 4 | 15 | |
| t ₆₆ CC | Other signals release | – | 20 | – | 20 | |
| t ₆₇ CC | Other signals drive | – 4 | 15 | – 4 | 15 | |

Figure 66. External bus arbitration (releasing the bus)



1. The ST10F276 will complete the currently running bus cycle before granting bus access.
2. This is the first possibility for $\overline{\text{BREQ}}$ to become active.
3. The $\overline{\text{CS}}$ outputs will be resistive high (pull-up) after t_{64} .

Figure 67. External bus arbitration (regaining the bus)

1. This is the last chance for \overline{BREQ} to trigger the indicated regain-sequence. Even if \overline{BREQ} is activated earlier, the regain-sequence is initiated by $HOLD$ going high. Please note that $HOLD$ may also be deactivated without the ST10F276 requesting the bus.
2. The next ST10F276 driven bus cycle may start here.

23.8.22 High-speed synchronous serial interface (SSC) timing modes

Master mode

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ\text{C}$, $C_L = 50\text{pF}$

Table 108. Master mode

| Symbol | Parameter | Max. baud rate 6.6MBd ⁽¹⁾ @F _{CPU} = 40 MHz (<SSCBR> = 0002h) | | Variable baud rate (<SSCBR> = 0001h - FFFFh) | | Unit |
|----------------------|--|--|------|--|------------|------|
| | | Min. | Max. | Min. | Max. | |
| t ₃₀₀ CC | SSC clock cycle time ⁽²⁾ | 150 | 150 | 8TCL | 262144 TCL | ns |
| t ₃₀₁ CC | SSC clock high time | 63 | – | t ₃₀₀ / 2 – 12 | – | |
| t ₃₀₂ CC | SSC clock low time | | | | | |
| t ₃₀₃ CC | SSC clock rise time | – | 10 | – | 10 | |
| t ₃₀₄ CC | SSC clock fall time | | 15 | | 15 | |
| t ₃₀₅ CC | Write data valid after shift edge | | | | | |
| t ₃₀₆ CC | Write data hold after shift edge | – 2 | – | – 2 | – | |
| t _{307p} SR | Read data setup time before latch edge, phase error detection on (SSCPEN = 1) | 37.5 | | 2TCL + 12.5 | | |
| t _{308p} SR | Read data hold time after latch edge, phase error detection on (SSCPEN = 1) | 50 | | 4TCL | | |
| t ₃₀₇ SR | Read data setup time before latch edge, phase error detection off (SSCPEN = 0) | 25 | | 2TCL | | |
| t ₃₀₈ SR | Read data hold time after latch edge, phase error detection off (SSCPEN = 0) | 0 | | 0 | | |

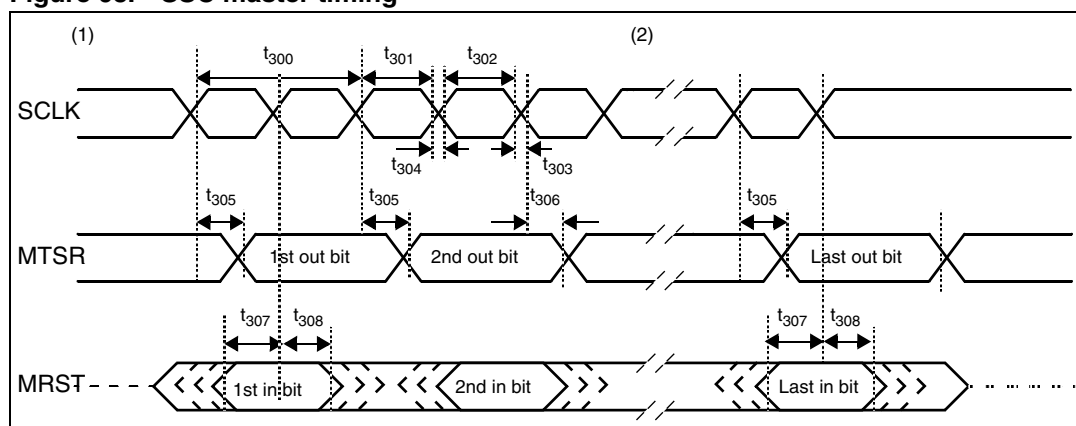
1. Maximum baud rate is in reality 8Mbaud, that can be reached with 64 MHz CPU clock and $\langle SSCBR \rangle$ set to '3h', or with 48 MHz CPU clock and $\langle SSCBR \rangle$ set to '2h'. When 40 MHz CPU clock is used the maximum baud rate cannot be higher than 6.6Mbaud ($\langle SSCBR \rangle = '2h'$) due to the limited granularity of $\langle SSCBR \rangle$. Value '1h' for $\langle SSCBR \rangle$ may be used only with CPU clock equal to (or lower than) 32 MHz (after checking that timings are in line with the target slave).

2. Formula for SSC Clock Cycle time:

$$t_{300} = 4 \text{ TCL} \times (\langle SSCBR \rangle + 1)$$

Where $\langle SSCBR \rangle$ represents the content of the SSC baud rate register, taken as unsigned 16-bit integer. Minimum limit allowed for t_{300} is 125ns (corresponding to 8Mbaud)

Figure 68. SSC master timing



1. The phase and polarity of shift and latch edge of SCLK is programmable. This figure uses the leading clock edge as shift edge (drawn in bold), with latch on trailing edge (SSCPH = 0b), idle clock line is low, leading clock edge is low-to-high transition (SSCPO = 0b).
2. The bit timing is repeated for all bits to be transmitted or received.

Slave mode

$V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $T_A = -40$ to $+125^\circ C$, $C_L = 50pF$

Table 109. Slave mode

| Symbol | Parameter | Max. baud rate 6.6 MBd ⁽¹⁾ @F _{CPU} = 40 MHz (<SSCBR> = 0002h) | | Variable baud rate (<SSCBR> = 0001h - FFFFh) | | Unit |
|----------------------|--|---|------|--|------------|------|
| | | Min. | Max. | Min. | Max. | |
| t ₃₁₀ SR | SSC clock cycle time ⁽²⁾ | 150 | 150 | 8TCL | 262144 TCL | ns |
| t ₃₁₁ SR | SSC clock high time | 63 | – | t ₃₁₀ / 2 – 12 | – | |
| t ₃₁₂ SR | SSC clock low time | | | | | |
| t ₃₁₃ SR | SSC clock rise time | – | 10 | – | 10 | |
| t ₃₁₄ SR | SSC clock fall time | | | | | |
| t ₃₁₅ CC | Write data valid after shift edge | | 55 | | 2TCL + 30 | |
| t ₃₁₆ CC | Write data hold after shift edge | 0 | – | 0 | – | |
| t _{317p} SR | Read data setup time before latch edge, phase error detection on (SSCPEN = 1) | 62 | | 4TCL + 12 | | |
| t _{318p} SR | Read data hold time after latch edge, phase error detection on (SSCPEN = 1) | 87 | | 6TCL + 12 | | |
| t ₃₁₇ SR | Read data setup time before latch edge, phase error detection off (SSCPEN = 0) | 6 | | 6 | | |
| t ₃₁₈ SR | Read data hold time after latch edge, phase error detection off (SSCPEN = 0) | 31 | | 2TCL + 6 | | |

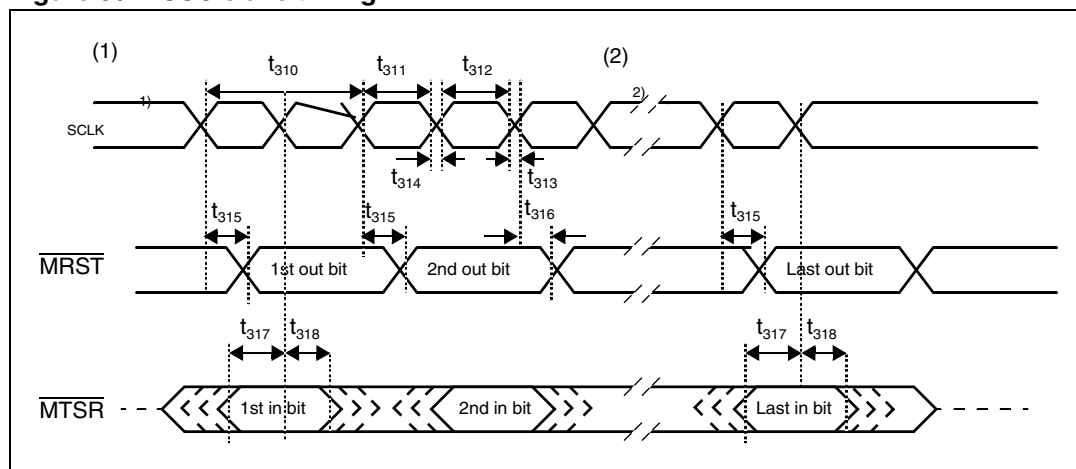
1. Maximum baud rate is in reality 8Mbaud, that can be reached with 64 MHz CPU clock and <SSCBR> set to '3h', or with 48 MHz CPU clock and <SSCBR> set to '2h'. When 40 MHz CPU clock is used the maximum baud rate cannot be higher than 6.6Mbaud (<SSCBR> = '2h') due to the limited granularity of <SSCBR>. Value '1h' for <SSCBR> may be used only with CPU clock lower than 32 MHz (after checking that timings are in line with the target master).

2. Formula for SSC Clock Cycle time:

$$t_{310} = 4 \text{ TCL} * (<\text{SSCBR}> + 1)$$

Where <SSCBR> represents the content of the SSC baud rate register, taken as unsigned 16-bit integer. Minimum limit allowed for t₃₁₀ is 125ns (corresponding to 8Mbaud).

Figure 69. SSC slave timing



1. The phase and polarity of shift and latch edge of SCLK is programmable. This figure uses the leading clock edge as shift edge (drawn in bold), with latch on trailing edge (SSCPH = 0b), idle clock line is low, leading clock edge is low-to-high transition (SSCPO = 0b).
2. The bit timing is repeated for all bits to be transmitted or received.

24 Package information

Figure 70. 144-pin plastic quad flat package

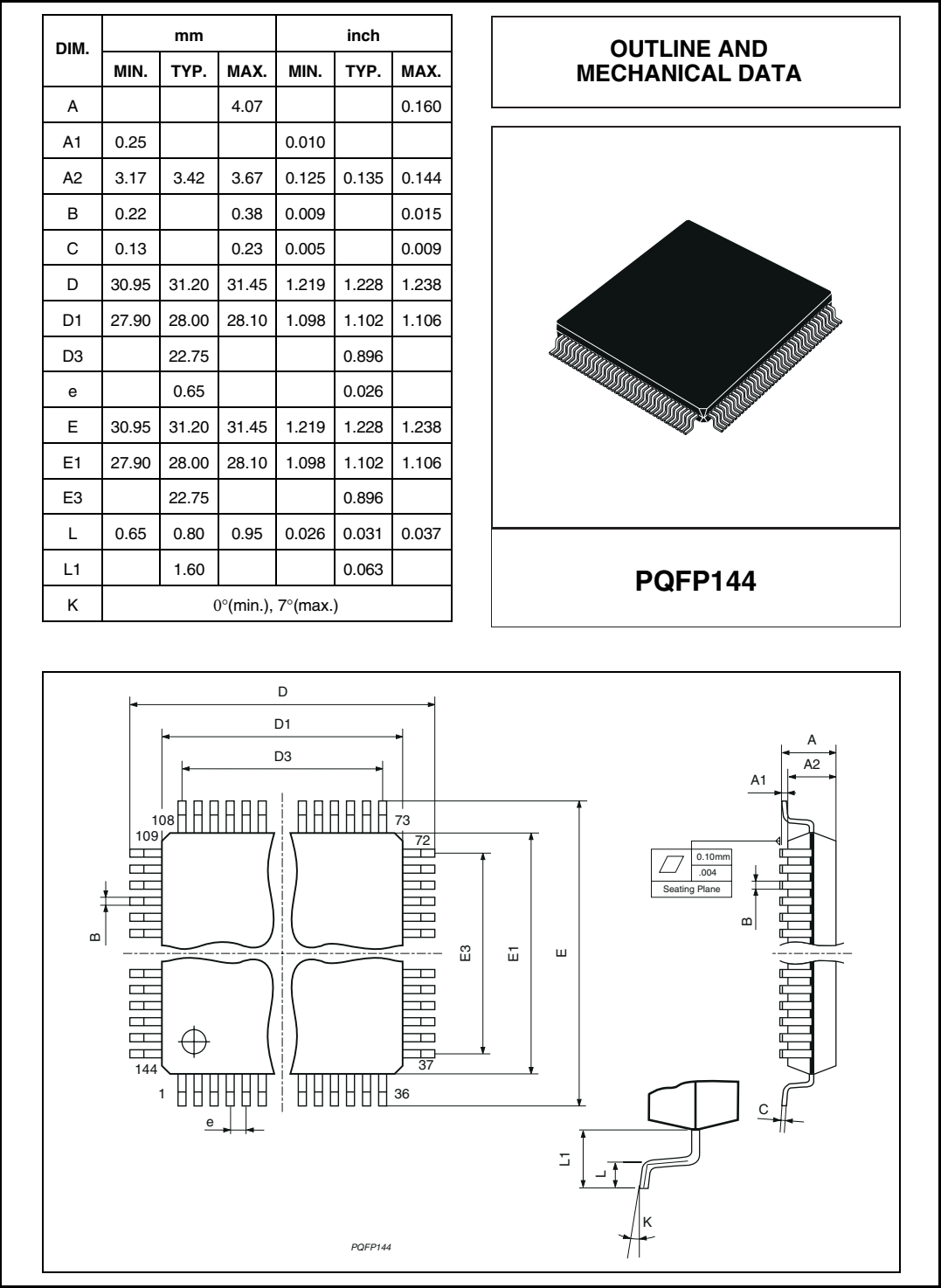
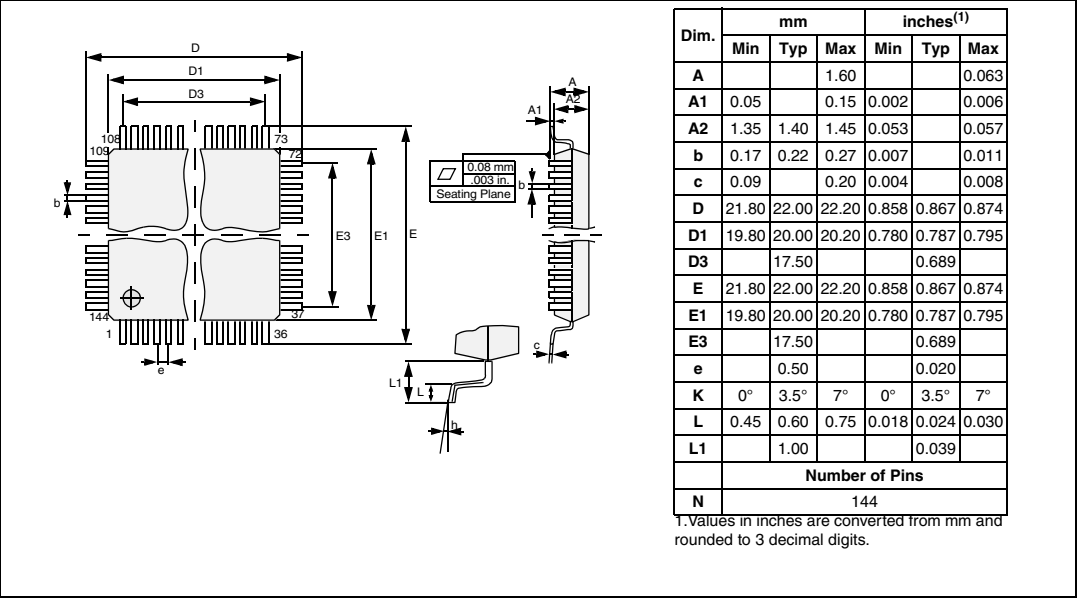


Figure 71. 144-pin low profile quad flat package (10x10)



25 Revision history

Table 110. Document revision history

| Date | Revision | Changes |
|--------------|----------|------------------|
| 02-June-2006 | 1 | Initial release. |

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST's terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST'S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED REPRESENTATIVE OF ST, ST PRODUCTS ARE NOT DESIGNED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS, WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2006 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com