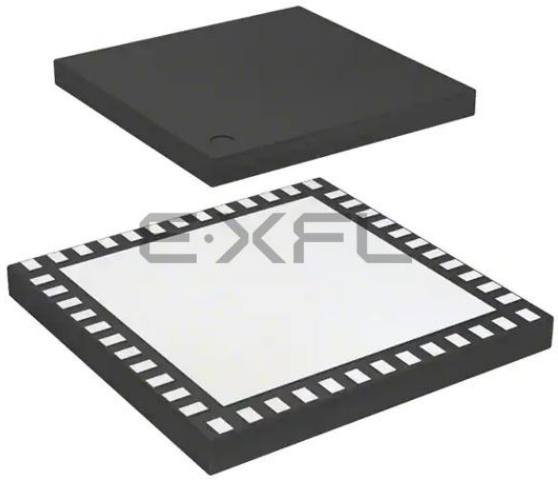


Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	AVR
Core Size	32-Bit Single-Core
Speed	50MHz
Connectivity	I²C, SPI, UART/USART
Peripherals	Brown-out Detect/Reset, DMA, POR, PWM, WDT
Number of I/O	36
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	32K x 8
Voltage - Supply (Vcc/Vdd)	1.62V ~ 3.6V
Data Converters	A/D 8x12b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	48-UFLGA Exposed Pad
Supplier Device Package	48-TLLGA (5.5x5.5)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/at32uc3l0128-d3hr

- Two Master and Two Slave Two-wire Interfaces (TWI), 400 kbit/s I^2C-compatible
- One 8-channel Analog-to-digital Converter (ADC) with up to 12 Bits Resolution
  - Internal Temperature Sensor
- Eight Analog Comparators (AC) with Optional Window Detection
- Capacitive Touch (CAT) Module
  - Hardware-assisted Atmel® AVR® QTouch® and Atmel® AVR® QMatrix Touch Acquisition
  - Supports QTouch and QMatrix Capture from Capacitive Touch Sensors
- QTouch Library Support
  - Capacitive Touch Buttons, Sliders, and Wheels
  - QTouch and QMatrix Acquisition
- On-chip Non-intrusive Debug System
  - Nexus Class 2+, Runtime Control, Non-intrusive Data and Program Trace
  - aWire Single-pin Programming Trace and Debug Interface Muxed with Reset Pin
  - NanoTrace Provides Trace Capabilities through JTAG or aWire Interface
- 48-pin TQFP/QFN/TLLGA (36 GPIO Pins)
- Five High-drive I/O Pins
- Single 1.62-3.6 V Power Supply

The page buffer is not automatically reset after a page write. The programmer should do this manually by issuing the Clear Page Buffer flash command. This can be done after a page write, or before the page buffer is loaded with data to be stored to the flash page.

8.5 Flash Commands

The FLASHCDW offers a command set to manage programming of the flash memory, locking and unlocking of regions, and full flash erasing. See Section 8.8.2 for a complete list of commands.

To run a command, the CMD field in the Flash Command Register (FCMD) has to be written with the command number. As soon as the FCMD register is written, the FRDY bit in the Flash Status Register (FSR) is automatically cleared. Once the current command is complete, the FSR.FRDY bit is automatically set. If an interrupt has been enabled by writing a one to FCR.FRDY, the interrupt request line of the Flash Controller is activated. All flash commands except for Quick Page Read (QPR) and Quick User Page Read (QPRUP) will generate an interrupt request upon completion if FCR.FRDY is one.

Any HSB bus transfers attempting to read flash memory when the FLASHCDW is busy executing a flash command will be stalled, and allowed to continue when the flash command is complete.

After a command has been written to FCMD, the programming algorithm should wait until the command has been executed before attempting to read instructions or data from the flash or writing to the page buffer, as the flash will be busy. The waiting can be performed either by polling the Flash Status Register (FSR) or by waiting for the flash ready interrupt. The command written to FCMD is initiated on the first clock cycle where the HSB bus interface in FLASHCDW is IDLE. The user must make sure that the access pattern to the FLASHCDW HSB interface contains an IDLE cycle so that the command is allowed to start. Make sure that no bus masters such as DMA controllers are performing endless burst transfers from the flash. Also, make sure that the CPU does not perform endless burst transfers from flash. This is done by letting the CPU enter sleep mode after writing to FCMD, or by polling FSR for command completion. This polling will result in an access pattern with IDLE HSB cycles.

All the commands are protected by the same keyword, which has to be written in the eight highest bits of the FCMD register. Writing FCMD with data that does not contain the correct key and/or with an invalid command has no effect on the flash memory; however, the PROGE bit is set in the Flash Status Register (FSR). This bit is automatically cleared by a read access to the FSR register.

Writing a command to FCMD while another command is being executed has no effect on the flash memory; however, the PROGE bit is set in the Flash Status Register (FSR). This bit is automatically cleared by a read access to the FSR register.

If the current command writes or erases a page in a locked region, or a page protected by the BOOTPROT fuses, the command has no effect on the flash memory; however, the LOCKE bit is set in the FSR register. This bit is automatically cleared by a read access to the FSR register.

8.5.1 Write/Erase Page Operation

Flash technology requires that an erase must be done before programming. The entire flash can be erased by an Erase All command. Alternatively, pages can be individually erased by the Erase Page command.

The User page can be written and erased using the mechanisms described in this chapter.

12.7.1 Main Clock Control

Name: MCCTRL

Access Type: Read/Write

Offset: 0x000

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-
7	6	5	4	3	2	1	0
-	-	-	-	-	MCSEL		

- **MCSEL: Main Clock Select**

Table 12-8. Main clocks in AT32UC3L0128/256.

MCSEL[2:0]	Main clock source
0	System RC oscillator (RCSYS)
1	Oscillator0 (OSC0)
2	DFLL
3	120MHz RC oscillator (RC120M)(1)

Note: 1. If the 120MHz RC oscillator is selected as main clock source, it must be divided by at least 4 before being used as clock source for the CPU. This division is selected by writing to the CPUSEL and CPUDIV bits in the CPUSEL register, before switching to RC120M as main clock source.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

12.7.4 PBx Clock Select

Name: PBxSEL

Access Type: Read/Write

Offset: 0x00C-0x010

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
PBDIV	-	-	-	-	PBSEL		

- **PBDIV, PBSEL: PBx Division and Clock Select**
  PBDIV = 0: PBx clock equals main clock.
  PBDIV = 1: PBx clock equals main clock divided by $2^{(PBSEL+1)}$.

Note that if PBDIV is written to 0, PBSEL should also be written to 0 to ensure correct operation.

Also note that writing this register clears SR.CKRDY. The register must not be re-written until SR.CKRDY goes high.

Note that this register is protected by a lock. To write to this register the UNLOCK register has to be written first. Please refer to the UNLOCK register description for details.

13.6.16 BOD Control Register

Name: BOD

Access Type: Read/Write

Reset Value: -

31	30	29	28	27	26	25	24
SFV	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	FCD

15	14	13	12	11	10	9	8
-	-	-	-	-	-	CTRL	

7	6	5	4	3	2	1	0
-	HYST	LEVEL					

- **SFV: Store Final Value**
  0: The register is read/write
  1: The register is read-only, to protect against further accidental writes.
  This bit is cleared after any reset except for a BOD reset, and during flash calibration.
- **FCD: Fuse Calibration Done**
  0: The flash calibration will be redone after any reset.
  1: The flash calibration will be redone after any reset except for a BOD reset.
  This bit is cleared after any reset, except for a BOD reset.
  This bit is set when the CTRL, HYST and LEVEL fields have been updated by the flash fuses after a reset.
- **CTRL: BOD Control**

Table 13-5. Operation Mode for BOD

CTRL	Description
0	BOD is disabled.
1	BOD is enabled and can reset the device. An interrupt request will be generated, if enabled in the IMR register.
2	BOD is enabled but cannot reset the device. An interrupt request will be generated, if enabled in the IMR register.
3	Reserved.

- **HYST: BOD Hysteresis**
  0: No hysteresis.
  1: Hysteresis on.
- **LEVEL: BOD Level**
  This field sets the triggering threshold of the BOD. See Electrical Characteristics for actual voltage levels.
  Note that any change to the LEVEL field of the BOD register should be done with the BOD deactivated to avoid spurious reset or interrupt.

14.6.13 Clock Control Register

Name: CLOCK

Access Type: Read/Write

Offset: 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	-	-	-	-	CSSEL		

7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	CEN

When writing to this register, follow the sequence in Section 14.5.1 on page 266.

- **CSSEL: Clock Source Selection**
  This field defines the clock source CLK_AST_PRSC for the prescaler:

Table 14-2. Clock Source Selection

CSSEL	Clock Source
0	System RC oscillator (RCSYS)
1	32KHz oscillator (OSC32K)
2	PB clock
3	Generic clock (GCLK)
4	1KHz clock from 32KHz oscillator (CLK_1K)

- **CEN: Clock Enable**
  0: CLK_AST_PRSC is disabled.
  1: CLK_AST_PRSC is enabled.

14.6.14 Digital Tuner Register

Name: DTR

Access Type: Read/Write

Offset: 0x44

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
-	-	ADD	EXP				

When the SR.BUSY bit is set writes to this register will be discarded and this register will read as zero.

- **VALUE:**
  0: The frequency is unchanged.
  1-255: The frequency will be adjusted according to the formula below.

- **ADD:**
  0: The resulting frequency is $f0\left(1 - \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} + 1}\right)$ for $VALUE > 0$.

  1: The resulting frequency is $f0\left(1 + \frac{1}{roundup\left(\frac{256}{VALUE}\right) \cdot 2^{(EXP)} - 1}\right)$ for $VALUE > 0$.

- **EXP:**
  The frequency will be adjusted according to the formula above.

17.4.2 Clocks

The clock for the FREQM bus interface (CLK_FREQM) is generated by the Power Manager. This clock is enabled at reset, and can be disabled in the Power Manager. It is recommended to disable the FREQM before disabling the clock, to avoid freezing the FREQM ia an undefined state.

A set of clocks can be selected as reference (CLK_REF) and another set of clocks can be selected for measurement (CLK_MSR). Please refer to the CLKSEL and REFSEL tables in the Module Configuration section for details.

17.4.3 Debug Operation

When an external debugger forces the CPU into debug mode, the FREQM continues normal operation. If the FREQM is configured in a way that requires it to be periodically serviced by the CPU through interrupts or similar, improper operation or data loss may result during debugging.

17.4.4 Interrupts

The FREQM interrupt request line is connected to the internal source of the interrupt controller. Using the FREQM interrupt requires the interrupt controller to be programmed first.

17.5 Functional Description

The FREQM accuratly measures the frequency of a clock by comparing the frequency to a known frequency:

$$f_{CLK\_MSR} = (VALUE/REFNUM)*f_{CLK\_REF}$$

17.5.1 Reference Clock

The Reference Clock Selection (REFSEL) field in the Mode Register (MODE) selects the clock source for CLK_REF. The reference clock is enabled by writing a one to the Reference Clock Enable (REFCEN) bit in the Mode Register. This clock should have a known frequency.

CLK_REF needs to be disabled before switching to another clock. The RCLKBUSY bit in the Status Register (SR) indicates whether the clock is busy or not. This bit is set when the MODE.REFCEN bit is written.

To change CLK_REF:

- Write a zero to the MODE.REFCEN bit to disable the clock, without changing the other bits/fields in the Mode Register.
- Wait until the SR.RCLKBUSY bit reads as zero.
- Change the MODE.REFSEL field.
- Write a one to the MODE.REFCEN bit to enable the clock, without changing the other bits/fields in the Mode Register.
- Wait until the SR.RCLKBUSY bit reads as zero.

To enable CLK_REF:

- Write the correct value to the MODE.REFSEL field.
- Write a one to the MODE.REFCEN to enable the clock, without changing the other bits/fields in the Mode Register.
- Wait until the SR.RCLKBUSY bit reads as zero.

To disable CLK_REF:

19.7.10 Receiver Time-out Register

Name: RTOR

Access Type: Read-write

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	TO[16]

15	14	13	12	11	10	9	8
TO[15:8]							

7	6	5	4	3	2	1	0
TO[7:0]							

This register can only be written to if write protection is disabled, see ”Write Protect Mode Register” on page 424.

- **TO: Time-out Value**
  0: The receiver Time-out is disabled.
  1 - 131071: The receiver Time-out is enabled and the time-out delay is TO x bit period.
  Note that the size of the TO counter is device dependent, see the Module Configuration section.

- **SWRST: Software Reset**
  This bit will always read as 0.
  Writing a zero to this bit has no effect.
  Writing a one to this bit resets the TWIS.
- **STREN: Clock Stretch Enable**
  0: Disables clock stretching if RHR/THR buffer full/empty. May cause over/underrun.
  1: Enables clock stretching if RHR/THR buffer full/empty.
- **GCMATCH: General Call Address Match**
  0: Causes the TWIS not to acknowledge the General Call Address.
  1: Causes the TWIS to acknowledge the General Call Address.
- **SMATCH: Slave Address Match**
  0: Causes the TWIS not to acknowledge the Slave Address.
  1: Causes the TWIS to acknowledge the Slave Address.
- **SMEN: SMBus Mode Enable**
  0: Disables SMBus mode.
  1: Enables SMBus mode.
- **SEN: Slave Enable**
  0: Disables the slave interface.
  1: Enables the slave interface.

22.9.11 Status Clear Register

Name: SCR

Access Type: Write-only

Offset: 0x28

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
BTF	REP	STO	SMBDAM	SMBHHM	SMBALERTM	GCM	SAM

15	14	13	12	11	10	9	8
-	BUSERR	SMBPECERR	SMBTOUT	-	-	-	NAK

7	6	5	4	3	2	1	0
ORUN	URUN	-	-	TCOMP	-	-	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in SR and the corresponding interrupt request.

24.7.7 Channel Register C

Name: RC

Access Type: Read/Write

Offset: 0x1C + n * 0x40

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
RC[15:8]							

7	6	5	4	3	2	1	0
RC[7:0]							

- **RC: Register C**
  RC contains the Register C value in real time.

- **TC0XC0S: External Clock Signal 0 Selection**

TC0XC0S	Signal Connected to XC0
0	TCLK0
1	none
2	TIOA1
3	TIOA2

26.9.10 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x24

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-

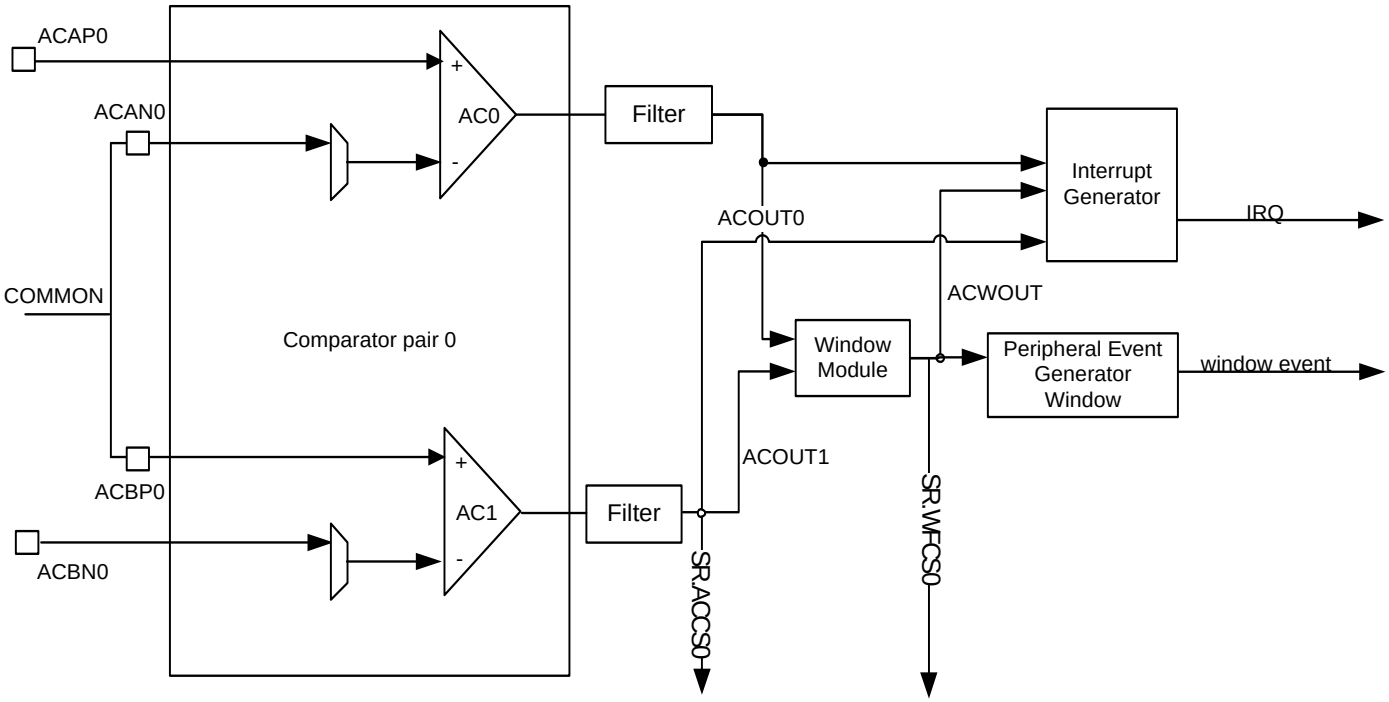
23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-

15	14	13	12	11	10	9	8
-	CELSE	CGT	CLT	-	-	BUSY	READY

7	6	5	4	3	2	1	0
-	-	NOCNT	PENCNT	-	-	OVRE	DRDY

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

Figure 27-2. Analog Comparator Interface in Window Mode



27.6.5.1 *Window Mode Output*

When operating in window mode, each channel generates the same ACOUT outputs as in normal mode, see Section 27.6.4.1.

Additionally, the ACIFB generates a window mode signal (acwout) according to the common input voltage to be compared:

- ACWOUT = 1 if the common input voltage is inside the window, $V_{ACN(N+1)} < V_{common} < V_{ACP(N)}$
- ACWOUT = 0 if the common input voltage is outside the window, $V_{common} < V_{ACN(N+1)}$ or $V_{common} > V_{ACP(N)}$
- ACWOUT = 0 if the window mode output is not available (SR.ACRDY*n*=0 or SR.ACRDY*n+1=0)*

27.6.5.2 *Window Mode Interrupts*

When operating in window mode, each channel can generate the same interrupts as in normal mode, see Section 27.6.4.2.

Additionally, when channels operate in window mode, programming Window Mode Interrupt Settings in the Window Mode Configuration Register (CONFWn.WIS) can cause interrupts to be generated when:

- As soon as the common input voltage is inside the window.
- As soon as the common input voltage is outside the window.
- On toggle of the window compare output (ACWOUT).
- When the comparison in both channels in the window pair is ready.

28.7.16 Interrupt Disable Register

Name: IDR

Access Type: Write-only

Offset: 0x48

Reset Value: -

31	30	29	28	27	26	25	24
-	-	-	-	-	-	-	-
23	22	21	20	19	18	17	16
DMATSC	-	-	-	-	-	-	-
15	14	13	12	11	10	9	8
-	-	-	-	-	-	ACQDONE	ACREADY
7	6	5	4	3	2	1	0
-	-	-	-	-	ATSC	ATCAL	-

Writing a zero to a bit in this register has no effect.
Writing a one to a bit in this register will clear the corresponding bit in IMR.

28.7.30 DMATouch Sensor Status Register

Name: DMATSS

Access Type: Read-only

Offset: 0xA0

Reset Value: 0x00000000

31	30	29	28	27	26	25	24
SS[31:24]							

23	22	21	20	19	18	17	16
SS[23:16]							

15	14	13	12	11	10	9	8
SS[15:8]							

7	6	5	4	3	2	1	0
SS[7:0]							

- **SS: Sensor Status**
  0: The DMATouch sensor is not active, i.e. the button is currently not pushed.
  1: The DMATouch sensor is active, i.e. the button is currently pushed.

1. The size of the data field: 7 (size and starting address + read length indicator) in the length field.
2. The size of the transfer: Words, halfwords, or bytes.
3. The starting address of the transfer.
4. The number of **bytes** to read (max 65532).

The 4 MSB of the 36 bit SAB address are submitted together with the size field (2 bits). The 4 remaining address bytes are submitted before the number of bytes to read. The size of the transfer is specified using the values from the following table:

Table 31-44. Size Field Decoding

Size field	Description
00	Byte transfer
01	Halfword transfer
10	Word transfer
11	Reserved

Below is an example read command:

1. 0x55 (sync)
2. 0x81 (command)
3. 0x00 (length MSB)
4. 0x07 (length LSB)
5. 0x25 (size and address MSB, the two MSB of this byte are unused and set to zero)
6. 0x00
7. 0x00
8. 0x00
9. 0x04 (address LSB)
10. 0x00
11. 0x04
12. 0xXX (CRC MSB)
13. 0xXX (CRC LSB)

The length field is set to 0x0007 because there are 7 bytes of additional data: 5 bytes of address and size and 2 bytes with the number of bytes to read. The address and size field indicates one word (four bytes) should be read from address 0x500000004.

Table 31-45. MEMORY_READ Details

Command	Details
Command value	0x81
Additional data	Size, Address and Length
Possible responses	0xC1: MEMDATA (Section 31.6.8.4) 0xC2: MEMORY_READWRITE_STATUS (Section 31.6.8.5) 0x41: NACK (Section 31.6.8.2)

31.6.7.11 HALT

This command tells the CPU to halt code execution for safe programming. If the CPU is not halted during programming it can start executing partially loaded programs. To halt the processor, the aWire master should send 0x01 in the data field of the command. After programming the halting can be released by sending 0x00 in the data field of the command.

Table 31-46. HALT Details

Command	Details
Command value	0x82
Additional data	0x01 to halt the CPU 0x00 to release the halt and reset the device.
Possible responses	0x40: ACK (Section 31.6.8.1) 0x41: NACK (Section 31.6.8.2)

31.6.7.12 RESET

This command resets different domains in the part. The aWire master sends a byte with the reset value. Each bit in the reset value byte corresponds to a reset domain in the chip. If a bit is set the reset is activated and if a bit is not set the reset is released. The number of reset domains and their destinations are identical to the resets described in the JTAG data registers chapter under reset register.

Table 31-47. RESET Details

Command	Details
Command value	0x83
Additional data	Reset value for each reset domain. The number of reset domains is part specific.
Possible responses	0x40: ACK (Section 31.6.8.1) 0x41: NACK (Section 31.6.8.2)

31.6.7.13 SET_GUARD_TIME

Sets the guard time value in the AW, i.e. how long the AW will wait before starting its transfer after the master has finished.

The guard time can be either 0x00 (128 bit lengths), 0x01 (16 bit lengths), 0x2 (4 bit lengths) or 0x3 (1 bit length).

Table 31-48. SET_GUARD_TIME Details

Command	Details
Command value	0x84
Additional data	Guard time
Possible responses	0x40: ACK (Section 31.6.8.1) 0x41: NACK (Section 31.6.8.2)

35.3.5 GPIO

1. **Clearing Interrupt flags can mask other interrupts**
   When clearing interrupt flags in a GPIO port, interrupts on other pins of that port, happening in the same clock cycle will not be registered.
   Fix/Workaround
   Read the PVR register of the port before and after clearing the interrupt to see if any pin change has happened while clearing the interrupt. If any change occurred in the PVR between the reads, they must be treated as an interrupt.

35.3.6 SPI

1. **SPI data transfer hangs with CSR0.CSAAT==1 and MR.MODFDIS==0**
   When CSR0.CSAAT==1 and mode fault detection is enabled (MR.MODFDIS==0), the SPI module will not start a data transfer.
   Fix/Workaround
   Disable mode fault detection by writing a one to MR.MODFDIS.

2. **Disabling SPI has no effect on the SR.TDRE bit**
   Disabling SPI has no effect on the SR.TDRE bit whereas the write data command is filtered when SPI is disabled. Writing to TDR when SPI is disabled will not clear SR.TDRE. If SPI is disabled during a PDCA transfer, the PDCA will continue to write data to TDR until its buffer is empty, and this data will be lost.
   Fix/Workaround
   Disable the PDCA, add two NOPs, and disable the SPI. To continue the transfer, enable the SPI and PDCA.

3. **SPI disable does not work in SLAVE mode**
   SPI disable does not work in SLAVE mode.
   Fix/Workaround
   Read the last received data, then perform a software reset by writing a one to the Software Reset bit in the Control Register (CR.SWRST).

4. **SPI bad serial clock generation on 2nd chip_select when SCBR=1, CPOL=1, and NCPHA=0**
   When multiple chip selects (CS) are in use, if one of the baudrates equal 1 while one (CSRn.SCBR=1) of the others do not equal 1, and CSRn.CPOL=1 and CSRn.NCPHA=0, then an additional pulse will be generated on SCK.
   Fix/Workaround
   When multiple CS are in use, if one of the baudrates equals 1, the others must also equal 1 if CSRn.CPOL=1 and CSRn.NCPHA=0.

5. **SPI mode fault detection enable causes incorrect behavior**
   When mode fault detection is enabled (MR.MODFDIS==0), the SPI module may not operate properly.
   Fix/Workaround
   Always disable mode fault detection before using the SPI by writing a one to MR.MODFDIS.

6. **SPI RDR.PCS is not correct**
   The PCS (Peripheral Chip Select) field in the SPI RDR (Receive Data Register) does not correctly indicate the value on the NPCS pins at the end of a transfer.
   Fix/Workaround
   Do not use the PCS field of the SPI RDR.