**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 | ARM® Cortex®-M3 |
| Core Size | 32-Bit Single-Core |
| Speed | 84MHz |
| Connectivity | CANbus, I²C, IrDA, LINbus, MMC, SPI, SSC, UART/USART, USB |
| Peripherals | Brown-out Detect/Reset, DMA, I²S, POR, PWM, WDT |
| Number of I/O | 63 |
| Program Memory Size | 256KB (256K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 64K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.62V ~ 3.6V |
| Data Converters | A/D 16x12b; D/A 2x12b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 100-TFBGA |
| Supplier Device Package | 100-TFBGA (9x9) |
| Purchase URL | https://www.e-xfl.com/product-detail/microchip-technology/atsam3a4ca-cu |

#### 10.4.3.14 Base Priority Mask Register

The BASEPRI register defines the minimum priority for exception processing. When BASEPRI is set to a nonzero value, it prevents the activation of all exceptions with same or lower priority level as the BASEPRI value. See the register summary in Table 10-2 on page 51 for its attributes. The bit assignments are:

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

- **BASEPRI**

Priority mask bits:

0x0000 = no effect

Nonzero = defines the base priority for exception processing.

The processor does not process any exception with a priority value greater than or equal to BASEPRI.

This field is similar to the priority fields in the interrupt priority registers. The processor implements only bits[7:4] of this field, bits[3:0] read as zero and ignore writes. See “Interrupt Priority Registers” on page 160 for more information. Remember that higher priority field values correspond to lower exception priorities.

### 10.13.6 MOV and MVN

Move and Move NOT.

#### 10.13.6.1 Syntax

```
MOV{S}{cond} Rd, Operand2
MOV{cond} Rd, #imm16
MVN{S}{cond} Rd, Operand2
```

where:

S is an optional suffix. If S is specified, the condition code flags are updated on the result of the operation, see “Conditional execution” on page 91.

cond is an optional condition code, see “Conditional execution” on page 91.

Rd is the destination register.

Operand2 is a flexible second operand. See “Flexible second operand” on page 86 for details of the options.

imm16 is any value in the range 0-65535.

#### 10.13.6.2 Operation

The MOV instruction copies the value of *Operand2* into *Rd*.

When *Operand2* in a MOV instruction is a register with a shift other than LSL #0, the preferred syntax is the corresponding shift instruction:

- ASR{S}{cond} Rd, Rm, #n is the preferred syntax for MOV{S}{cond} Rd, Rm, ASR #n
- LSL{S}{cond} Rd, Rm, #n is the preferred syntax for MOV{S}{cond} Rd, Rm, LSL #n if *n* != 0
- LSR{S}{cond} Rd, Rm, #n is the preferred syntax for MOV{S}{cond} Rd, Rm, LSR #n
- ROR{S}{cond} Rd, Rm, #n is the preferred syntax for MOV{S}{cond} Rd, Rm, ROR #n
- RRX{S}{cond} Rd, Rm is the preferred syntax for MOV{S}{cond} Rd, Rm, RRX.

Also, the MOV instruction permits additional forms of *Operand2* as synonyms for shift instructions:

- MOV{S}{cond} Rd, Rm, ASR Rs is a synonym for ASR{S}{cond} Rd, Rm, Rs
- MOV{S}{cond} Rd, Rm, LSL Rs is a synonym for LSL{S}{cond} Rd, Rm, Rs
- MOV{S}{cond} Rd, Rm, LSR Rs is a synonym for LSR{S}{cond} Rd, Rm, Rs
- MOV{S}{cond} Rd, Rm, ROR Rs is a synonym for ROR{S}{cond} Rd, Rm, Rs

See “ASR, LSL, LSR, ROR, and RRX” on page 114.

The MVN instruction takes the value of *Operand2*, performs a bitwise logical NOT operation on the value, and places the result into *Rd*.

The MOVW instruction provides the same function as MOV, but is restricted to using the *imm16* operand.

#### 10.13.6.3 Restrictions

You can use SP and PC only in the MOV instruction, with the following restrictions:

- the second operand must be a register without shift
- you must not specify the S suffix.

When *Rd* is PC in a MOV instruction:

- bit[0] of the value written to the PC is ignored
- a branch occurs to the address created by forcing bit[0] of that value to 0.

Though it is possible to use MOV as a branch instruction, ARM strongly recommends the use of a BX or BLX instruction to branch for software portability to the ARM instruction set.

### 10.23.1 MPU Type Register

The TYPE register indicates whether the MPU is present, and if so, how many regions it supports. See the register summary in Table 10-35 on page 198 for its attributes. The bit assignments are:

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

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

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

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | SEPARATE |

- **IREGION**

Indicates the number of supported MPU instruction regions.

Always contains 0x00. The MPU memory map is unified and is described by the DREGION field.

- **DREGION**

Indicates the number of supported MPU data regions:

0x08 = Eight MPU regions.

- **SEPARATE**

Indicates support for unified or separate instruction and date memory maps:

0: unified.

### 18.4.2 Read Operations

An optimized controller manages embedded Flash reads, thus increasing performance when the processor is running in Thumb2 mode by means of the 128- or 64- bit wide memory interface.

The Flash memory is accessible through 8-, 16- and 32-bit reads.

As the Flash block size is smaller than the address space reserved for the internal memory area, the embedded Flash wraps around the address space and appears to be repeated within it.

The read operations can be performed with or without wait states. Wait states must be programmed in the field FWS (Flash Read Wait State) in the Flash Mode Register (EEFC_FMR). Defining FWS to be 0 enables the single-cycle access of the embedded Flash. Refer to the Electrical Characteristics for more details.

#### 18.4.2.1 128-bit or 64-bit Access Mode

By default the read accesses of the Flash are performed through a 128-bit wide memory interface. It enables better system performance especially when 2 or 3 wait state needed.

For systems requiring only 1 wait state, or to privilege current consumption rather than performance, the user can select a 64-bit wide memory access via the FAM bit in the Flash Mode Register (EEFC_FMR)

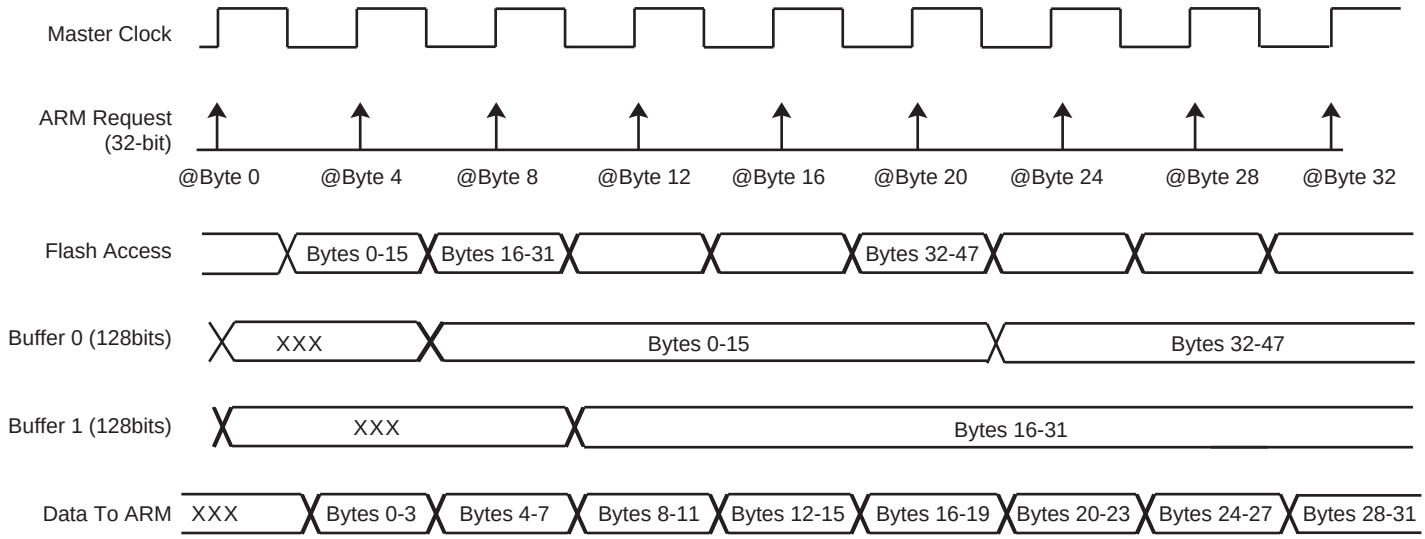
Please refer to the electrical characteristics section of the product datasheet for more details.

#### 18.4.2.2 Code Read Optimization

A system of 2 x 128-bit or 2 x 64-bit buffers is added in order to optimize sequential Code Fetch.

Note: Immediate consecutive code read accesses are not mandatory to benefit from this optimization.

**Figure 18-2. Code Read Optimization for FWS = 0**




Note: When FWS is equal to 0, all the accesses are performed in a single-cycle access.

### 22.7.16 DMAC Channel x [x = 0..5] Control A Register

**Name:** DMAC_CTRLAx [x = 0..5]

**Address:** 0x400C4048 [0], 0x400C4070 [1], 0x400C4098 [2], 0x400C40C0 [3], 0x400C40E8 [4], 0x400C4110 [5]

**Access:** Read-write

**Reset:** 0x00000000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| DONE | – | DST_WIDTH | | – | – | SRC_WIDTH | |

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| – | DCSIZE | | | – | SCSIZE | | |

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

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BTSIZE | | | | | | | |

This register can only be written if the WPEN bit is cleared in “DMAC Write Protect Mode Register” on page 380

- **BTSIZE: Buffer Transfer Size**

The transfer size relates to the number of transfers to be performed, that is, for writes it refers to the number of source width transfers to perform when DMAC is flow controller. For Reads, BTSIZE refers to the number of transfers completed on the Source Interface. When this field is set to 0, the DMAC module is automatically disabled when the relevant channel is enabled.

- **SCSIZE: Source Chunk Transfer Size.**

| Value | Name | Description |
|---|---|---|
| 000 | CHK_1 | 1 data transferred |
| 001 | CHK_4 | 4 data transferred |
| 010 | CHK_8 | 8 data transferred |
| 011 | CHK_16 | 16 data transferred |
| 100 | CHK_32 | 32 data transferred |
| 101 | CHK_64 | 64 data transferred |
| 110 | CHK_128 | 128 data transferred |
| 111 | CHK_256 | 256 data transferred |

## 25.10 Standard Read and Write Protocols

In the following sections, the byte access type is not considered. Byte select lines (NBS0 to NBS1) always have the same timing as the A address bus. NWE represents either the NWE signal in byte select access type or one of the byte write lines (NWR0 to NWR1) in byte write access type. NWR0 to NWR3 have the same timings and protocol as NWE. In the same way, NCS represents one of the NCS[0..7] chip select lines.

### 25.10.1 Read Waveforms

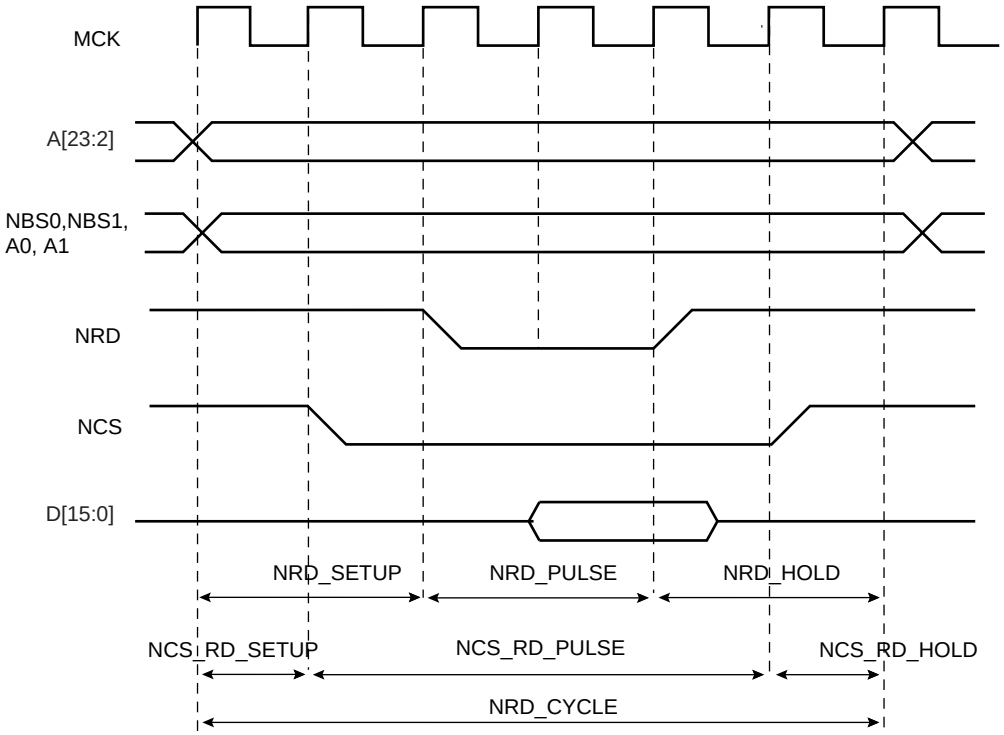
The read cycle is shown on Figure 25-7.

The read cycle starts with the address setting on the memory address bus, i.e.:

{A[23:2], A1, A0} for 8-bit devices

{A[23:2], A1} for 16-bit devices

**Figure 25-7. Standard Read Cycle**




#### 25.10.1.1 NRD Waveform

The NRD signal is characterized by a setup timing, a pulse width and a hold timing.

1. NRD_SETUP: the NRD setup time is defined as the setup of address before the NRD falling edge.
2. NRD_PULSE: the NRD pulse length is the time between NRD falling edge and NRD rising edge.
3. NRD_HOLD: the NRD hold time is defined as the hold time of address after the NRD rising edge.

### 28.15.9 PMC Clock Generator Main Clock Frequency Register

**Name:** CKGR_MCFR

**Address:** 0x400E0624

**Access:** Read-only

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

| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | MAINFRDY |

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

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MAINF | | | | | | | |

This register can only be written if the WPEN bit is cleared in “PMC Write Protect Mode Register” .

- **MAINF: Main Clock Frequency**

Gives the number of Main Clock cycles within 16 Slow Clock periods.

- **MAINFRDY: Main Clock Ready**

0 = MAINF value is not valid or the Main Oscillator is disabled .

1 = The Main Oscillator has been enabled previously and MAINF value is available.

## 33.7 Functional Description

### 33.7.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see Figure 33-4).

Each transfer begins with a START condition and terminates with a STOP condition (see Figure 33-3).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

**Figure 33-3. START and STOP Conditions**

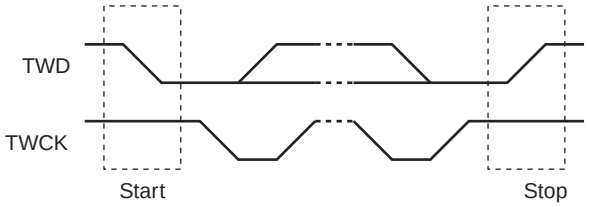
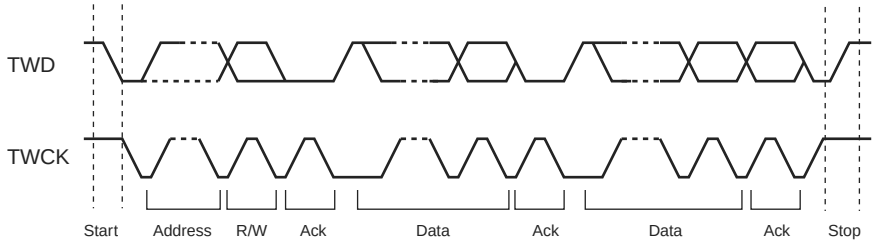



**Figure 33-4. Transfer Format**




### 33.7.2 Modes of Operation

The TWI has six modes of operations:

- Master transmitter mode
- Master receiver mode
- Multi-master transmitter mode
- Multi-master receiver mode
- Slave transmitter mode
- Slave receiver mode

These modes are described in the following chapters.

### 34.6.6 UART Status Register

**Name:** UART_SR

**Address:** 0x400E0814

**Access:** Read-only

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| – | – | – | – | – | – | – | – |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| – | – | – | RXBUFF | TXBUFE | – | TXEMPTY | – |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PARE | FRAME | OVRE | ENDTX | ENDRX | – | TXRDY | RXRDY |

- **RXRDY: Receiver Ready**

0 = No character has been received since the last read of the UART_RHR or the receiver is disabled.

1 = At least one complete character has been received, transferred to UART_RHR and not yet read.

- **TXRDY: Transmitter Ready**

0 = A character has been written to UART_THR and not yet transferred to the Shift Register, or the transmitter is disabled.

1 = There is no character written to UART_THR not yet transferred to the Shift Register.

- **ENDRX: End of Receiver Transfer**

0 = The End of Transfer signal from the receiver Peripheral Data Controller channel is inactive.

1 = The End of Transfer signal from the receiver Peripheral Data Controller channel is active.

- **ENDTX: End of Transmitter Transfer**

0 = The End of Transfer signal from the transmitter Peripheral Data Controller channel is inactive.

1 = The End of Transfer signal from the transmitter Peripheral Data Controller channel is active.

- **OVRE: Overrun Error**

0 = No overrun error has occurred since the last RSTSTA.

1 = At least one overrun error has occurred since the last RSTSTA.

- **FRAME: Framing Error**

0 = No framing error has occurred since the last RSTSTA.

1 = At least one framing error has occurred since the last RSTSTA.

- **PARE: Parity Error**

0 = No parity error has occurred since the last RSTSTA.

1 = At least one parity error has occurred since the last RSTSTA.

- **TXEMPTY: Transmitter Empty**

0 = There are characters in UART_THR, or characters being processed by the transmitter, or the transmitter is disabled.

1 = There are no characters in UART_THR and there are no characters being processed by the transmitter.

- **LINTC: LIN Transfer Completed Interrupt Enable**
- **CTSIC: Clear to Send Input Change Interrupt Enable**
- **MANE: Manchester Error Interrupt Enable**
- **LINBE: LIN Bus Error Interrupt Enable**
- **LINISFE: LIN Inconsistent Synch Field Error Interrupt Enable**
- **LINIPE: LIN Identifier Parity Interrupt Enable**
- **LINCE: LIN Checksum Error Interrupt Enable**
- **LINSNRE: LIN Slave Not Responding Error Interrupt Enable**

- **ACPC: RC Compare Effect on TIOA**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

- **AEEVT: External Event Effect on TIOA**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

- **ASWTRG: Software Trigger Effect on TIOA**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

- **BCPB: RB Compare Effect on TIOB**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

- **BCPC: RC Compare Effect on TIOB**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

- **BEEVT: External Event Effect on TIOB**

| Value | Name | Description |
|---|---|---|
| 0 | NONE | None |
| 1 | SET | Set |
| 2 | CLEAR | Clear |
| 3 | TOGGLE | Toggle |

### 37.14.11 HSMCI Transmit Data Register

**Name:** HSMCI_TDR

**Address:** 0x40000034

**Access:** Write-only

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

- **DATA: Data to Write**

# 38. Pulse Width Modulation (PWM)

## 38.1 Description

The PWM macrocell controls 8 channels independently. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM master clock (MCK).

All PWM macrocell accesses are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the Peripheral DMA Controller Channel (PDC) which offers buffer transfer without processor Intervention.

The PWM macrocell provides 8 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs), and to trigger PDC transfer requests.

The PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM block provides a fault protection mechanism with 6 fault inputs**,** capable of detecting a fault condition and to override the PWM outputs asynchronously.

For safety usage, some control registers are write-protected.

# 39. USB On-The-Go Interface (UOTGHS)

## 39.1 Description

The Universal Serial Bus (USB) MCU device complies with the Universal Serial Bus (USB) 2.0 specification in all speeds.

Each pipe/endpoint can be configured in one of several USB transfer types. It can be associated with one, two or three banks of a DPRAM used to store the current data payload. If two or three banks are used, then one DPRAM bank is read or written by the CPU or the DMA, while the other is read or written by the UOTGHS core. This feature is mandatory for isochronous pipes/endpoints.

Table 39-1 on page 1053 describes the hardware configuration of the USB MCU device.

**Table 39-1. Description of USB Pipes/Endpoints**

| Pipe/Endpoint | Mnemonic | Max. Nb. Banks | DMA | High Band Width | Max. Pipe/ Endpoint Size | Type |
|---|---|---|---|---|---|---|
| 0 | PEP_0 | 1 | N | N | 64 | Control |
| 1 | PEP_1 | 3 | Y | 1 | 1024 | Isochronous/Bulk/Interrupt/Control |
| 2 | PEP_2 | 3 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 3 | PEP_3 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 4 | PEP_4 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 5 | PEP_5 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 6 | PEP_6 | 2 | Y | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 7 | PEP_7 | 2 | N | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 8 | PEP_8 | 2 | _ | Y | 1024 | Isochronous/Bulk/Interrupt/Control |
| 9 | PEP_9 | 2 | _ | Y | 1024 | Isochronous/Bulk/Interrupt/Control |

Note: Bit fields are presented throughout the document as follows: ‘REGISTER. FIELD’ (e.g. UOTGHS_CTRL.USBE)

## 39.2 Embedded Characteristics

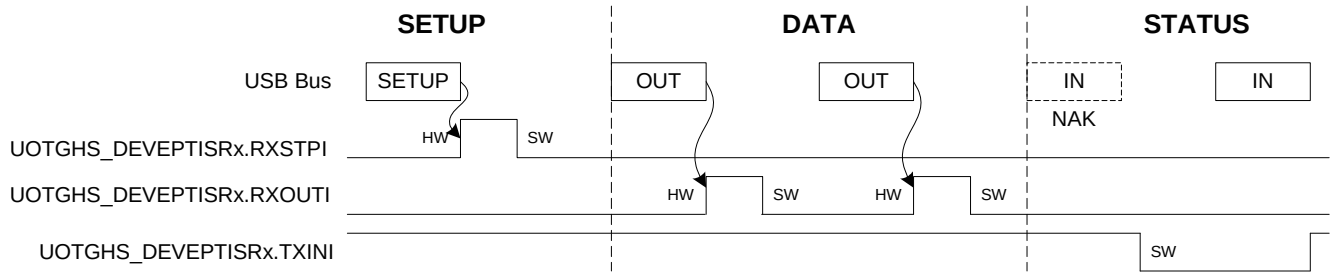
- Compatible with the USB 2.0 specification
- Supports High (480 Mbps), Full (12 Mbps) and Low (1.5 Mbps) speed communication and On-The-Go
- 10 pipes/endpoints
- 4096 bytes of Embedded Dual-Port RAM (DPRAM) for Pipes/Endpoints
- Up to 3 memory banks per Pipe/Endpoint (Not for Control Pipe/Endpoint)
- Flexible Pipe/Endpoint configuration and management with dedicated DMA channels
- On-Chip UTMI transceiver including Pull-Ups/Pull-downs
- On-Chip OTG pad including VBUS analog comparator

*Control write*

Figure 39-14 on page 1070 shows a control write transaction. During the status stage, the controller will not necessarily send a NAK on the first IN token:

- if the user knows the exact number of descriptor bytes that must be read, it can then anticipate the status stage and send a zero-length packet after the next IN token, or
- it can read the bytes and wait for the NAKed IN Interrupt (UOTGHS_DEVEPTISRx.NAKINI), which tells that all the bytes have been sent by the host and that the transaction is now in the status stage.

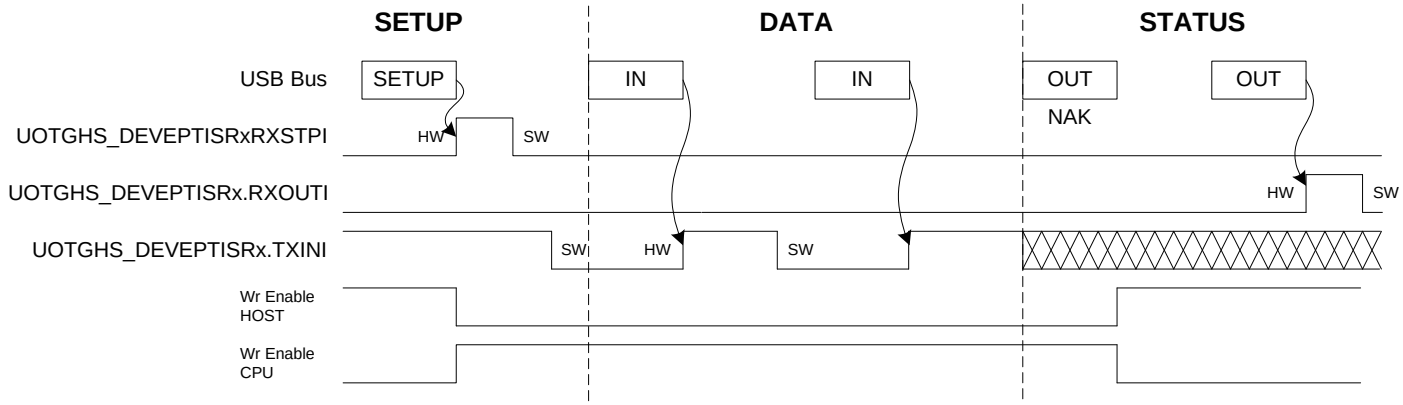
**Figure 39-14. Control Write**




*Control read*

Figure 39-15 on page 1070 shows a control read transaction. The UOTGHS has to manage the simultaneous write requests from the CPU and the USB host.

**Figure 39-15. Control Read**




A NAK handshake is always generated on the first status stage command.

When the controller detects the status stage, all data written by the CPU is lost and clearing UOTGHS_DEVEPTISRx.TXINI has no effect.

The user checks if the transmission or the reception is complete.

The OUT retry is always ACKed. This reception sets UOTGHS_DEVEPTISRx.RXOUTI and UOTGHS_DEVEPTISRx.TXINI. Handle this with the following software algorithm:

```
set TXINI
wait for RXOUTI OR TXINI
if RXOUTI, then clear bit and return
if TXINI, then continue
```

- **HBISOFLUSHIS: High Bandwidth Isochronous IN Flush Interrupt Set**

Writing a one to this bit will set **HBISOFLUSHI** in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **NAKOUTIS: NAKed OUT Interrupt Set**

Writing a one to this bit will set NAKOUTI in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **HBISOINERRIS: High bandwidth isochronous IN Underflow Error Interrupt Set**

Writing a one to this bit will set **HBISOINERRI** in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **RXSTPIS: Received SETUP Interrupt Set**

Writing a one to this bit will set RXSTPI in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **UNDERFIS: Underflow Interrupt Set**

Writing a one to this bit will set UNDERFI in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **RXOUTIS: Received OUT Data Interrupt Set**

Writing a one to this bit will set RXOUTI in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **TXINIS: Transmitted IN Data Interrupt Set**

Writing a one to this bit will set TXINI in UOTGHS_DEVEPTISRx, which may be useful for test or debug purposes.

Writing a zero to this bit has no effect.

This bit always reads as zero.

#### 39.6.3.18 Host Pipe x Disable Register

**Name:** UOTGHS_HSTPIPIDRx [x=0..9]

**Address:** 0x400AC620

**Access:** Write-only

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | – |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| – | – | – | – | – | – | PFREEZEC | PDISHDMAC |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| – | FIFOCONC | – | NBUSYBKEC | – | – | – | – |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SHORT PACKETIEC | RXSTALLDEC/ CRCERREC | OVERFIEC | NAKEDEC | PERREC | TXSTPEC/ UNDERFIEC | TXOUTEC | RXINEC |

- **PFREEZEC: Pipe Freeze Disable**

Writing a one to this bit will clear PFREEZE bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **PDISHDMAC: Pipe Interrupts Disable HDMA Request Disable**

Writing a one to this bit will clear PDISHDMA bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **FIFOCONC: FIFO Control Disable**

Writing a one to this bit will clear FIFOCON bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **NBUSYBKEC: Number of Busy Banks Disable**

Writing a one to this bit will clear NBUSYBKE bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **SHORTPACKETIEC: Short Packet Interrupt Disable**

Writing a one to this bit will clear SHORTPACKETIE bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **RXSTALLDEC: Received STALLed Interrupt Disable**

Writing a one to this bit will clear RXSTALLDE bit in UOTGHS_HSTPIPIMRx.

Writing a zero to this bit has no effect.

This bit always reads as zero.

- **TSTP: Timestamp Interrupt Mask**

0: TSTP interrupt is disabled.

1: TSTP interrupt is enabled.

- **CERR: CRC Error Interrupt Mask**

0: CRC Error interrupt is disabled.

1: CRC Error interrupt is enabled.

- **SERR: Stuffing Error Interrupt Mask**

0: Bit Stuffing Error interrupt is disabled.

1: Bit Stuffing Error interrupt is enabled.

- **AERR: Acknowledgment Error Interrupt Mask**

0: Acknowledgment Error interrupt is disabled.

1: Acknowledgment Error interrupt is enabled.

- **FERR: Form Error Interrupt Mask**

0: Form Error interrupt is disabled.

1: Form Error interrupt is enabled.

- **BERR: Bit Error Interrupt Mask**

0: Bit Error interrupt is disabled.

1: Bit Error interrupt is enabled.

### 45.10.6 SMC Timings

SMC timings are given for the following domains:

- 1.8V domain: $V_{DDIO}$ from 1.65V to 1.95V, maximum external capacitor = 30 pF
- 3.3V domain: $V_{DDIO}$ from 3.0V to 3.6V, maximum external capacitor = 50 pF

Timings are given assuming a capacitance load on data, control and address pads.

In the following tables $t_{CPMCK}$ is MCK period.

#### 45.10.6.1 Read Timings

**Table 45-50. SMC Read Signals - NRD Controlled (READ_MODE = 1)**

| Symbol | Parameter | Min | | Max | | Unit |
|---|---|---|---|---|---|---|
| | VDDIO supply | 1.8V Domain | 3.3V Domain | 1.8V Domain | 3.3V Domain | |
| **NO HOLD SETTINGS (NRD_HOLD = 0)** | | | | | | |
| $SMC_1$ | Data Setup before NRD High | 22.5 | 18 | | | ns |
| $SMC_2$ | Data Hold after NRD High | 0 | 0 | | | ns |
| **HOLD SETTINGS (NRD_HOLD ≠ 0)** | | | | | | |
| $SMC_3$ | Data Setup before NRD High | 20.3 | 16 | | | ns |
| $SMC_4$ | Data Hold after NRD High | 0 | 0 | | | ns |
| **HOLD or NO HOLD SETTINGS (NRD_HOLD ≠ 0, NRD_HOLD = 0)** | | | | | | |
| $SMC_5$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High | (NRD_SETUP + NRD_PULSE) × $t_{CPMCK}$ - 20 | (NRD_SETUP + NRD_PULSE) × $t_{CPMCK}$ - 20 | | | ns |
| $SMC_6$ | NCS low before NRD High | (NRD_SETUP + NRD_PULSE - NCS_RD_SETUP) × $t_{CPMCK}$ - 20 | (NRD_SETUP + NRD_PULSE - NCS_RD_SETUP) × $t_{CPMCK}$ - 20 | | | ns |
| $SMC_7$ | NRD Pulse Width | nrd pulse × $t_{CPMCK}$ - 3 | nrd pulse × $t_{CPMCK}$ - 3 | | | ns |

**Table 45-51. SMC Read Signals - NCS Controlled (READ_MODE = 0)**

| Symbol | Parameter | Min | | Max | | Unit |
|---|---|---|---|---|---|---|
| | VDDIO supply | 1.8V Domain | 3.3V Domain | 1.8V Domain | 3.3V Domain | |
| **NO HOLD SETTINGS (NCS_RD_HOLD = 0)** | | | | | | |
| $SMC_8$ | Data Setup before NCS High | 26.3 | 24.6 | | | ns |
| $SMC_9$ | Data Hold after NCS High | 0 | 0 | | | ns |
| **HOLD SETTINGS (NCS_RD_HOLD ≠ 0)** | | | | | | |
| $SMC_{10}$ | Data Setup before NCS High | 24.1 | 22.4 | | | ns |
| $SMC_{11}$ | Data Hold after NCS High | 0 | 0 | | | ns |
| **HOLD or NO HOLD SETTINGS (NCS_RD_HOLD ≠ 0, NCS_RD_HOLD = 0)** | | | | | | |
| $SMC_{12}$ | NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE) × $t_{CPMCK}$ - 3 | (NCS_RD_SETUP + NCS_RD_PULSE) × $t_{CPMCK}$ - 3 | | | ns |
| $SMC_{13}$ | NRD low before NCS High | (NCS_RD_SETUP + NCS_RD_PULSE - NRD_SETUP) × $t_{CPMCK}$ + 0.7 | (NCS_RD_SETUP + NCS_RD_PULSE - NRD_SETUP) × $t_{CPMCK}$ + 0.6 | | | ns |
| $SMC_{14}$ | NCS Pulse Width | NCS_RD_PULSE length × $t_{CPMCK}$ - 6 | NCS_RD_PULSE length × $t_{CPMCK}$ - 6 | | | ns |

**Table 45-57. USART SPI Timings**

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| **Master Mode** | | | | | |
| $SPI_0$ | $t_{CPSCK}$ Period | 1.8V domain<br>3.3V domain | $t_{CPMCK}$ / 6 | | ns |
| $SPI_1$ | Input Data Setup Time | 1.8V domain<br>3.3V domain | $0.5 \times t_{CPMCK}$ + 7.7 | | ns |
| $SPI_2$ | Input Data Hold Time | 1.8V domain<br>3.3V domain | $1.5 \times t_{CPMCK}$ + 1.8<br>$1.5 \times t_{CPMCK}$ + 0.7 | | ns |
| $SPI_3$ | Chip Select Active to Serial Clock | 1.8V domain<br>3.3V domain | $1.5 \times t_{CPSCK}$ -1 | | ns |
| $SPI_4$ | Output Data Setup Time | 1.8V domain<br>3.3V domain | 0 | 7.4 | ns |
| $SPI_5$ | Serial Clock to Chip Select Inactive | 1.8V domain<br>3.3V domain | | $1 \times t_{CPSCK}$ - 1 | ns |
| **Slave Mode** | | | | | |
| $SPI_6$ | $t_{CPSCK}$ falling to MISO | 1.8V domain<br>3.3V domain | 7 | 30 | ns |
| $SPI_7$ | MOSI Setup time before $t_{CPSCK}$ rises | 1.8V domain<br>3.3V domain | $2 \times t_{CPMCK}$ + 6.8 | | ns |
| $SPI_8$ | MOSI Hold time after $t_{CPSCK}$ rises | 1.8V domain<br>3.3V domain | 1 | | ns |
| $SPI_9$ | $t_{CPSCK}$ rising to MISO | 1.8V domain<br>3.3V domain | 7 | | ns |
| $SPI_{10}$ | MOSI Setup time before $t_{CPSCK}$ falls | 1.8V domain<br>3.3V domain | $2 \times t_{CPMCK}$ + 6 | | ns |
| $SPI_{11}$ | MOSI Hold time after $t_{CPSCK}$ falls | 1.8V domain<br>3.3V domain | 2.7<br>1 | | ns |
| $SPI_{12}$ | NPCS0 setup to $t_{CPSCK}$ rising | 1.8V domain<br>3.3V domain | $2.5 \times t_{CPMCK}$ +1 | | ns |
| $SPI_{13}$ | NPCS0 hold after $t_{CPSCK}$ falling | 1.8V domain<br>3.3V domain | $1.5 \times t_{CPMCK}$ + 4 | | ns |
| $SPI_{14}$ | NPCS0 setup to $t_{CPSCK}$ falling | 1.8V domain<br>3.3V domain | $2.5 \times t_{CPMCK}$ + 0.4 | | ns |
| $SPI_{15}$ | NPCS0 hold after $t_{CPSCK}$ rising | 1.8V domain<br>3.3V domain | $1.5 \times t_{CPMCK}$ + 2.7 | | ns |