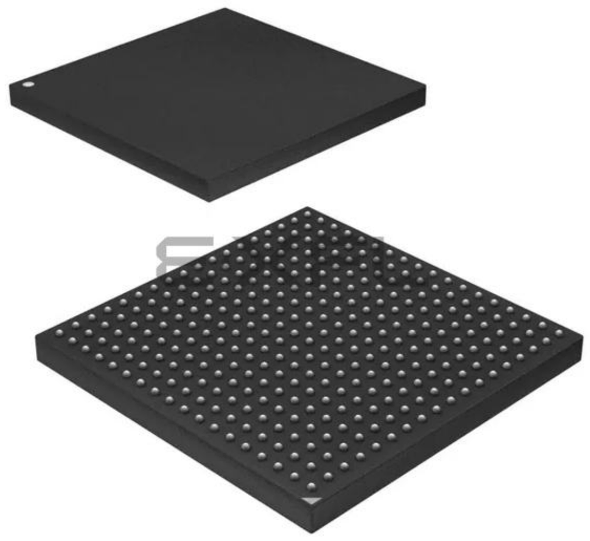
Microchip Technology - ATSAMA5D36A-CN Datasheet



Welcome to E-XFL.COM

Understanding Embedded - Microprocessors

Embedded microprocessors are specialized computing chips designed to perform specific tasks within an embedded system. Unlike general-purpose microprocessors found in personal computers, embedded microprocessors are tailored for dedicated functions within larger systems, offering optimized performance, efficiency, and reliability. These microprocessors are integral to the operation of countless electronic devices, providing the computational power necessary for controlling processes, handling data, and managing communications.

Applications of Embedded - Microprocessors

Embedded microprocessors are utilized across a broad spectrum of applications, making them indispensable in

Details	
Product Status	Active
Core Processor	ARM® Cortex®-A5
Number of Cores/Bus Width	1 Core, 32-Bit
Speed	536MHz
Co-Processors/DSP	-
RAM Controllers	LPDDR, LPDDR2, DDR2
Graphics Acceleration	No
Display & Interface Controllers	LCD, Touchscreen
Ethernet	10/100/1000Mbps (1)
SATA	-
USB	USB 2.0 (3)
Voltage - I/O	1.2V, 1.8V, 3.3V
Operating Temperature	-40°C ~ 105°C (TA)
Security Features	AES, SHA, TDES, TRNG
Package / Case	324-LFBGA
Supplier Device Package	324-LFBGA (15x15)
Purchase URL	https://www.e-xfl.com/product-detail/microchip-technology/atsama5d36a-cn

4.2 324-ball LFBGA Package Pinout

Table 4-1. SAMA5D3 Pinout for 324-ball LFBGA Package

			Primary		Alternate		PIO Peripheral A		PIO Peripheral B		PIO Peripheral C		Reset State
Pin	Power Rail	I/O Type	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal	Dir	Signal, Dir, PU, PD, HiZ, ST
E3	VDDIOP0	GPIO	PA0	I/O	—	—	LCDDAT0	O	—	—	—	—	PIO, I, PU, ST
F5	VDDIOP0	GPIO	PA1	I/O	—	—	LCDDAT1	O	—	—	—	—	PIO, I, PU, ST
D2	VDDIOP0	GPIO	PA2	I/O	—	—	LCDDAT2	O	—	—	—	—	PIO, I, PU, ST
F4	VDDIOP0	GPIO	PA3	I/O	—	—	LCDDAT3	O	—	—	—	—	PIO, I, PU, ST
D1	VDDIOP0	GPIO	PA4	I/O	—	—	LCDDAT4	O	—	—	—	—	PIO, I, PU, ST
J10	VDDIOP0	GPIO	PA5	I/O	—	—	LCDDAT5	O	—	—	—	—	PIO, I, PU, ST
G4	VDDIOP0	GPIO	PA6	I/O	—	—	LCDDAT6	O	—	—	—	—	PIO, I, PU, ST
J9	VDDIOP0	GPIO	PA7	I/O	—	—	LCDDAT7	O	—	—	—	—	PIO, I, PU, ST
F3	VDDIOP0	GPIO	PA8	I/O	—	—	LCDDAT8	O	—	—	—	—	PIO, I, PU, ST
J8	VDDIOP0	GPIO	PA9	I/O	—	—	LCDDAT9	O	—	—	—	—	PIO, I, PU, ST
E2	VDDIOP0	GPIO	PA10	I/O	—	—	LCDDAT10	O	—	—	—	—	PIO, I, PU, ST
K8	VDDIOP0	GPIO	PA11	I/O	—	—	LCDDAT11	O	—	—	—	—	PIO, I, PU, ST
F2	VDDIOP0	GPIO	PA12	I/O	—	—	LCDDAT12	O	—	—	—	—	PIO, I, PU, ST
G6	VDDIOP0	GPIO	PA13	I/O	—	—	LCDDAT13	O	—	—	—	—	PIO, I, PU, ST
E1	VDDIOP0	GPIO	PA14	I/O	—	—	LCDDAT14	O	—	—	—	—	PIO, I, PU, ST
H5	VDDIOP0	GPIO	PA15	I/O	—	—	LCDDAT15	O	—	—	—	—	PIO, I, PU, ST
H3	VDDIOP0	GPIO	PA16	I/O	—	—	LCDDAT16	O	—	—	ISI_D0	I	PIO, I, PU, ST
H6	VDDIOP0	GPIO	PA17	I/O	—	—	LCDDAT17	O	—	—	ISI_D1	I	PIO, I, PU, ST
H4	VDDIOP0	GPIO	PA18	I/O	—	—	LCDDAT18	O	TWD2	I/O	ISI_D2	I	PIO, I, PU, ST
H7	VDDIOP0	GPIO	PA19	I/O	—	—	LCDDAT19	O	TWCK2	O	ISI_D3	I	PIO, I, PU, ST
H2	VDDIOP0	GPIO	PA20	I/O	—	—	LCDDAT20	O	PWMH0	O	ISI_D4	I	PIO, I, PU, ST
J6	VDDIOP0	GPIO	PA21	I/O	—	—	LCDDAT21	O	PWML0	O	ISI_D5	I	PIO, I, PU, ST
G2	VDDIOP0	GPIO	PA22	I/O	—	—	LCDDAT22	O	PWMH1	O	ISI_D6	I	PIO, I, PU, ST
J5	VDDIOP0	GPIO	PA23	I/O	—	—	LCDDAT23	O	PWML1	O	ISI_D7	I	PIO, I, PU, ST
F1	VDDIOP0	GPIO	PA24	I/O	—	—	LCDPWM	O	—	—	—	—	PIO, I, PU, ST
J4	VDDIOP0	GPIO	PA25	I/O	—	—	LCDDISP	O	—	—	—	—	PIO, I, PU, ST
G3	VDDIOP0	GPIO	PA26	I/O	—	—	LCDVSYNC	O	—	—	—	—	PIO, I, PU, ST
J3	VDDIOP0	GPIO	PA27	I/O	—	—	LCDHSYNC	O	—	—	—	—	PIO, I, PU, ST
G1	VDDIOP0	GPIO_CLK2	PA28	I/O	—	—	LCDPCK	O	—	—	—	—	PIO, I, PU, ST
K4	VDDIOP0	GPIO	PA29	I/O	—	—	LCDDEN	O	—	—	—	—	PIO, I, PU, ST
H1	VDDIOP0	GPIO	PA30	I/O	—	—	TWD0	I/O	URXD1	I	ISI_VSYNC	I	PIO, I, PU, ST
K3	VDDIOP0	GPIO	PA31	I/O	—	—	TWCK0	O	UTXD1	O	ISI_HSYNC	I	PIO, I, PU, ST
T2	VDDIOP1	GMAC	PB0	I/O	—	—	GTX0	O	PWMH0	O	—	—	PIO, I, PU, ST
N7	VDDIOP1	GMAC	PB1	I/O	—	—	GTX1	O	PWML0	O	—	—	PIO, I, PU, ST
T3	VDDIOP1	GMAC	PB2	I/O	—	—	GTX2	O	TK1	I/O	—	—	PIO, I, PU, ST
N6	VDDIOP1	GMAC	PB3	I/O	—	—	GTX3	O	TF1	I/O	—	—	PIO, I, PU, ST
P5	VDDIOP1	GMAC	PB4	I/O	—	—	GRX0	I	PWMH1	O	—	—	PIO, I, PU, ST
T4	VDDIOP1	GMAC	PB5	I/O	—	—	GRX1	I	PWML1	O	—	—	PIO, I, PU, ST
R4	VDDIOP1	GMAC	PB6	I/O	—	—	GRX2	I	TD1	O	—	—	PIO, I, PU, ST
U1	VDDIOP1	GMAC	PB7	I/O	—	—	GRX3	I	RK1	I	—	—	PIO, I, PU, ST
R5	VDDIOP1	GMAC	PB8	I/O	—	—	GTXCK	I	PWMH2	O	—	—	PIO, I, PU, ST
P3	VDDIOP1	GMAC	PB9	I/O	—	—	GTXEN	O	PWML2	O	—	—	PIO, I, PU, ST

Note: Configuration registers such as AIC_SMR, AIC_SSR, return the values corresponding to the interrupt source selected by INTSEL.

The internal interrupt sources wired on the interrupt outputs of the embedded peripherals can be programmed either in level-sensitive mode or in edge-triggered mode. The active level of the internal interrupts is not important for the user.

The external interrupt sources can be programmed either in high level-sensitive or low level-sensitive modes, or in positive edge-triggered or negative edge-triggered modes.

17.8.1.2 Interrupt Source Enabling

Each interrupt source, including the FIQ in source 0, can be enabled or disabled by using the command registers; "AIC Interrupt Enable Command Register" on page 139 and "AIC Interrupt Disable Command Register" on page 140. The interrupt mask of the selected interrupt source can be read in the AIC_IMR register. A disabled interrupt does not affect servicing of other interrupts.

17.8.1.3 Interrupt Clearing and Setting

All interrupt sources programmed to be edge-triggered (including the FIQ in source 0) can be individually set or cleared by writing respectively the AIC_ISCR and AIC_ICCR registers. Clearing or setting interrupt sources programmed in level-sensitive mode has no effect.

The clear operation is perfunctory, as the software must perform an action to reinitialize the "memorization" circuitry activated when the source is programmed in edge-triggered mode. However, the set operation is available for auto-test or software debug purposes. It can also be used to execute an AIC-implementation of a software interrupt.

The AIC features an automatic clear of the current interrupt when the AIC_IVR (Interrupt Vector Register) is read. Only the interrupt source being detected by the AIC as the current interrupt is affected by this operation. (See "Priority Controller" on page 117.) The automatic clear reduces the operations required by the interrupt service routine entry code to reading the AIC_IVR. Note that the automatic interrupt clear is disabled if the interrupt source has the Fast Forcing feature enabled as it is considered uniquely as a FIQ source. (For further details, see Section 17.8.4.5 "Fast Forcing").

The automatic clear of the interrupt source 0 is performed when AIC_FVR is read.

17.8.1.4 Interrupt Status

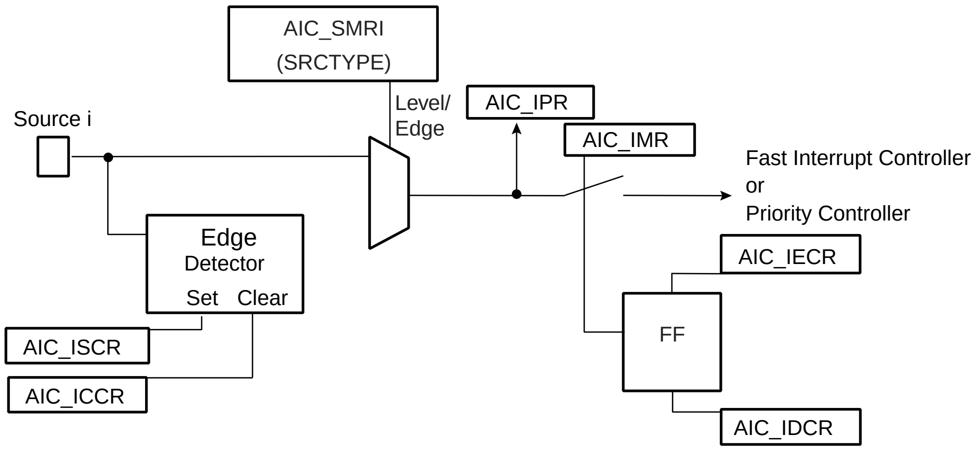
AIC_IPR registers represent the state of the interrupt lines, whether they are masked or not. The AIC_IMR register permits to define the mask of the interrupt lines.

The AIC_ISR register reads the number of the current interrupt (see "Priority Controller" on page 117) and the register AIC_CISR gives an image of the signals nIRQ and nFIQ driven on the processor.

Each status referred to above can be used to optimize the interrupt handling of the systems.

17.8.1.5 Internal Interrupt Source Input Stage

Figure 17-4. Internal Interrupt Source Input Stage



19.5.1 Reset Controller Control Register

Name: RSTC_CR

Address: 0xFFFFFE00

Access Type: Write-only

31	30	29	28	27	26	25	24
KEY							

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

15	14	13	12	11	10	9	8
–	–	–	–	–	–		–

7	6	5	4	3	2	1	0
–	–	–	–	EXTRST	PERRST	–	PROCRST

- **PROCRST: Processor Reset**

0: No effect

1: If KEY is correct, resets the processor

- **PERRST: Peripheral Reset**

0: No effect

1: If KEY is correct, resets the peripherals

- **EXTRST: External Reset**

0: No effect

1: If KEY is correct, asserts the NRST pin and resets the processor and the peripherals

- **KEY: Write Access Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

24. Slow Clock Controller (SCKC)

24.1 Description

The System Controller embeds a Slow Clock Controller.

The slow clock can be generated either by an external 32768 Hz crystal oscillator or by the on-chip 32 kHz RC oscillator. The 32768 Hz crystal oscillator can be bypassed by setting the OSC32BYP bit to accept an external slow clock on XIN32.

The internal 32 kHz RC oscillator and the 32768 Hz oscillator can be enabled by setting to 1, respectively, the RCEN and OSC32EN bits in the System Controller user interface. The OSCSEL command selects the slow clock source.

24.2 Embedded Characteristics

- 32 kHz RC Oscillator or 32768 Hz Crystal Oscillator Selector
- VDDBU Powered

reading the PIO Lock Status register (PIO_LOCKSR). Once an I/O line is locked, the only way to unlock it is to apply a hardware reset to the PIO Controller.

27.5.12 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0 to PA31. For any details, refer to the product electrical characteristics.

27.5.13 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. By default the Schmitt trigger is active. Disabling the Schmitt trigger is requested when using the QTouch™ Library.

27.5.14 Register Write Protection

To prevent any single software error from corrupting PIO behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the "PIO Write Protection Mode Register" (PIO_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the "PIO Write Protection Status Register" (PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the PIO_WPSR.

The following registers can be write-protected:

- "PIO Enable Register" on page 289
- "PIO Disable Register" on page 290
- "PIO Output Enable Register" on page 292
- "PIO Output Disable Register" on page 293
- "PIO Input Filter Enable Register" on page 295
- "PIO Input Filter Disable Register" on page 296
- "PIO Multi-driver Enable Register" on page 306
- "PIO Multi-driver Disable Register" on page 307
- "PIO Pull-Up Disable Register" on page 309
- "PIO Pull-Up Enable Register" on page 310
- "PIO Peripheral ABCD Select Register 1" on page 312
- "PIO Peripheral ABCD Select Register 2" on page 313
- "PIO Output Write Enable Register" on page 321
- "PIO Output Write Disable Register" on page 322
- "PIO Pad Pull-Down Disable Register" on page 318
- "PIO Pad Pull-Down Status Register" on page 320

27.6 I/O Lines Programming Example

The programming example shown in Table 27-1 is used to obtain the following configuration.

- 4-bit output port on I/O lines 0 to 3, (should be written in a single write operation), open-drain, with pull-up resistor
- Four output signals on I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor
- Four input signals on I/O lines 8 to 11 (to read push-button states for example), with pull-up resistors, glitch filters and input change interrupts
- Four input signals on I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter
- I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor
- I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor

28.1.3 MPDDR Controller Block Diagram

Figure 28-2. Organization of the MPDDRC

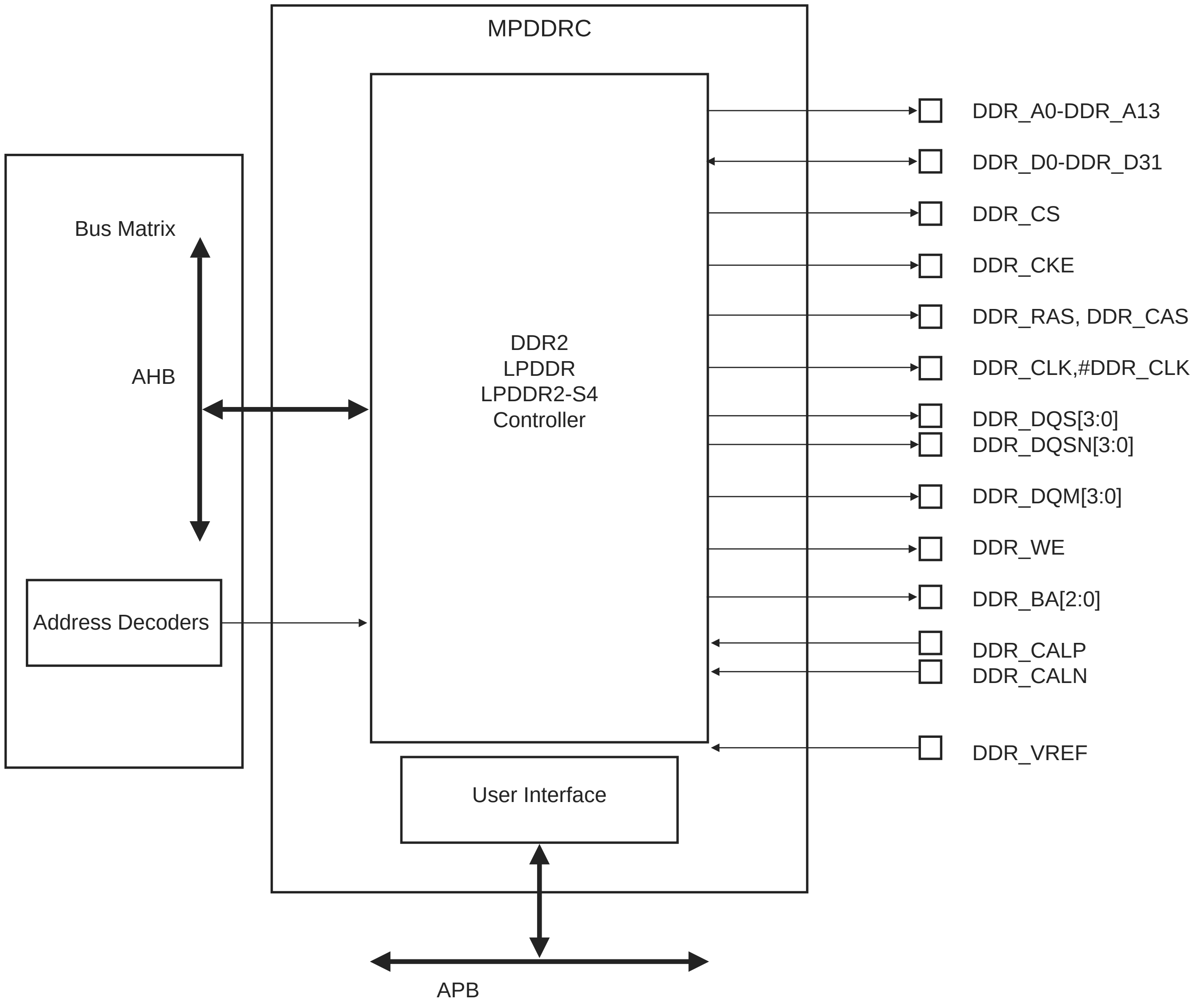
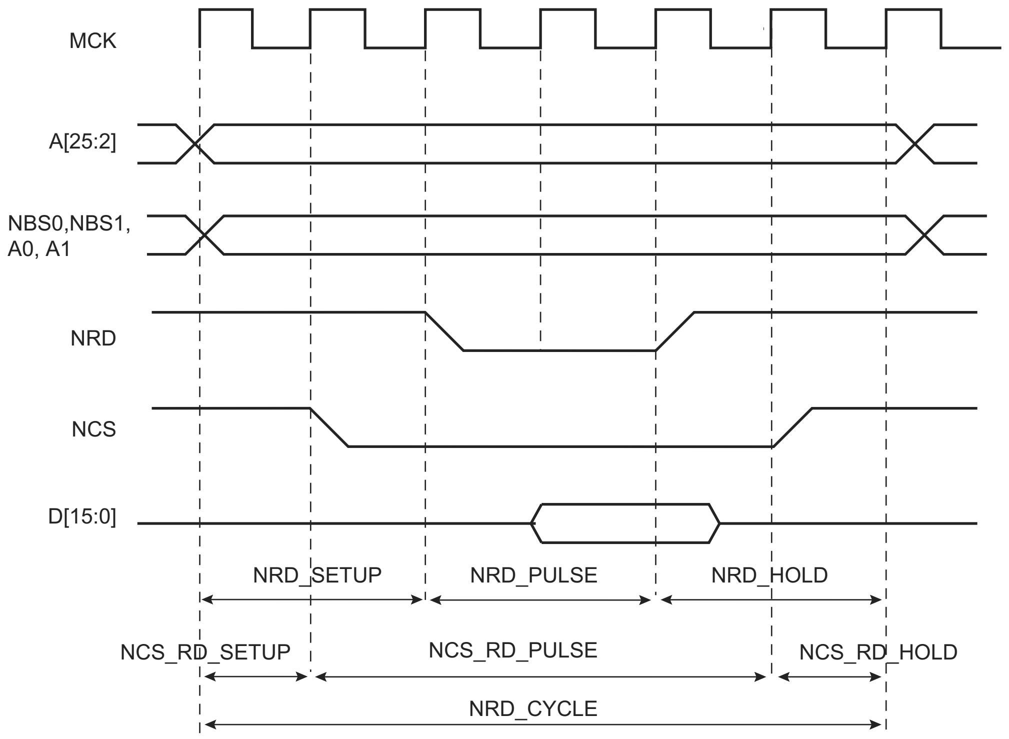


Figure 30-7. Standard Read Cycle



30.10.1.1 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.

Table 30-19. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x14*CS_number+0x604	HSMC Pulse Register	HSMC_PULSE	Read/Write	–
0x14*CS_number+0x608	HSMC Cycle Register	HSMC_CYCLE	Read/Write	–
0x14*CS_number+0x60C	HSMC Timings Register	HSMC_TIMINGS	Read/Write	–
0x14*CS_number+0x610	HSMC Mode Register	HSMC_MODE	Read/Write	–
0x6A0	HSMC Off Chip Memory Scrambling Register	HSMC_OCMS	Read/Write	0x0
0x6A4	HSMC Off Chip Memory Scrambling KEY1 Register	HSMC_KEY1	Write-once	0x0
0x6A8	HSMC Off Chip Memory Scrambling KEY2 Register	HSMC_KEY2	Write-once	0x0
0x6AC–0x6E0	Reserved	–	–	–
0x6E4	HSMC Write Protection Mode Register	HSMC_WPMR	Read/Write	0x0
0x6E8	HSMC Write Protection Status Register	HSMC_WPSR	Read-only	0x0
0x6FC	Reserved	–	–	–

30.20.38 HSMC Off Chip Memory Scrambling Key1 Register

Name: HSMC_KEY1

Address: 0xFFFFC6A4

Access: Write-once

Reset: 0x00000000

31	30	29	28	27	26	25	24
KEY1							

23	22	21	20	19	18	17	16
KEY1							

15	14	13	12	11	10	9	8
KEY1							

7	6	5	4	3	2	1	0
KEY1							

- **KEY1: Off Chip Memory Scrambling (OCMS) Key Part 1**

When Off Chip Memory Scrambling is enabled by setting the HSMC_OMCS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY1 and KEY2 values.

32.7.101High End Overlay Layer Configuration 6 Register

Name:	LCDC_HEOCFG6
Address:	0xF00303A4
Access:	Read-write
Reset:	0x00000000

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

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

32.7.121High End Overlay Layer Configuration 26 Register

Name: LCDC_HEOCFG26

Address: 0xF00303F4

Access: Read-write

Reset: 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**
XPHI4COEFF4							

- **XPHI4COEFF4: Horizontal Coefficient for phase 4 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

33.5.10 ISI Control Register

Name:	ISI_CR
Address:	0xF0034024
Access:	Write-only
Reset:	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
–	–	–	–	–	–	–	ISI_CDC

7	6	5	4	3	2	1	0
–	–	–	–	–	ISI_SRST	ISI_DIS	ISI_EN

- **ISI_EN: ISI Module Enable Request**

Write a one to this bit to enable the module. Software must poll ENABLE bit in the ISI_SR to verify that the command has successfully completed.

- **ISI_DIS: ISI Module Disable Request**

Write a one to this bit to disable the module. If both ISI_EN and ISI_DIS are asserted at the same time, the disable request is not taken into account. Software must poll DIS_DONE bit in the ISI_SR to verify that the command has successfully completed.

- **ISI_SRST: ISI Software Reset Request**

Write a one to this bit to request a software reset of the module. Software must poll SRST bit in the ISI_SR to verify that the software request command has terminated.

- **ISI_CDC: ISI Codec Request**

Write a one to this bit to enable the codec datapath and capture a full resolution frame. A new request cannot be taken into account while CDC_PND bit is active in the ISI_SR.

37.6.22 Specific Address 4 Bottom Register

Name: EMAC_SA4B

Address: 0xF802C0B0

Access: Read-write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

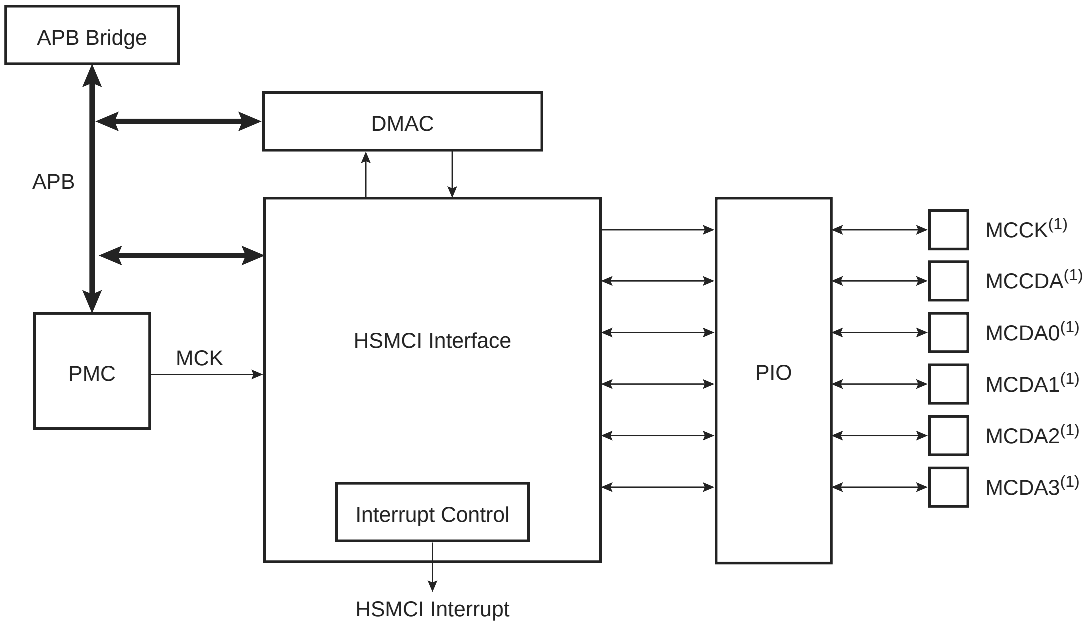
15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR**

Least significant bits of the destination address. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

Figure 38-2. Block Diagram (4-bit configuration)



Note: 1. When several HSMCI (x HSMCI) are embedded in a product, MCCK refers to HSMCIx_CK, MCCDA to HSMCIx_CDA, MCDAy to HSMCIx_DAy.

39.8.4 SPI Transmit Data Register

Name:	SPI_TDR
Address:	0xF000400C (0), 0xF800800C (1)
Access:	Write-only

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

23	22	21	20	19	18	17	16
–	–	–	–	PCS			

15	14	13	12	11	10	9	8
TD							

7	6	5	4	3	2	1	0
TD							

- **TD: Transmit Data**

Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

- **PCS: Peripheral Chip Select**

This field is only used if Variable Peripheral Select is active (PS = 1).

If PCSDEC = 0:

PCS = xxx0	NPCS[3:0] = 1110
PCS = xx01	NPCS[3:0] = 1101
PCS = x011	NPCS[3:0] = 1011
PCS = 0111	NPCS[3:0] = 0111
PCS = 1111	forbidden (no peripheral is selected)

(x = don’t care)

If PCSDEC = 1:

NPCS[3:0] output signals = PCS

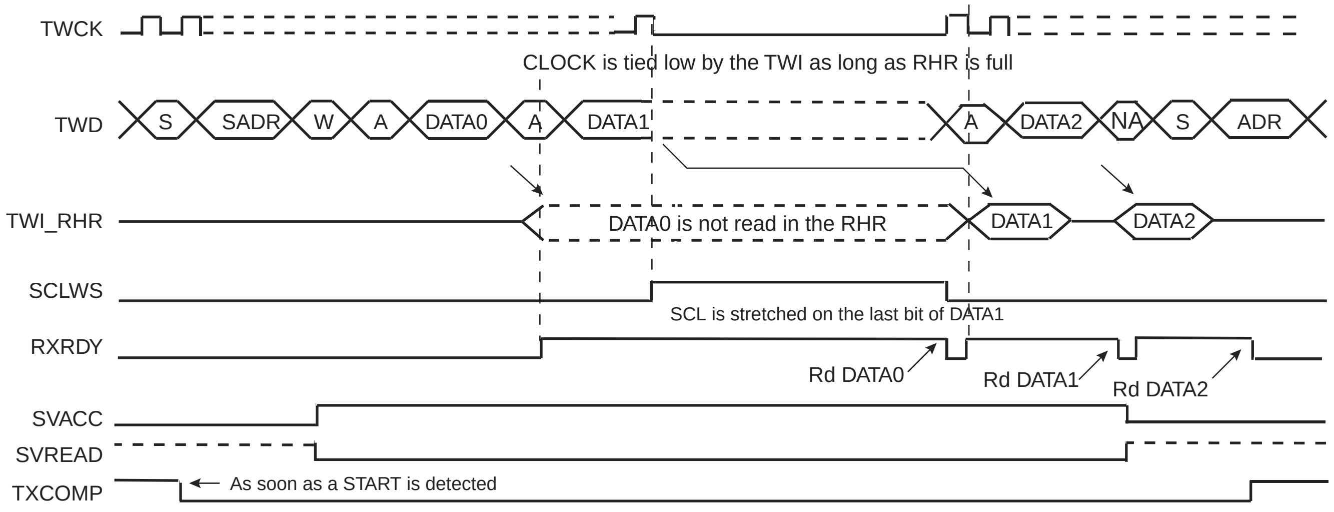
- **LASTXFER: Last Transfer**

0 = No effect.

1 = The current NPCS will be deasserted after the character written in TD has been transferred. When CSAAT is set, this allows to close the communication with the current serial peripheral by raising the corresponding NPCS line as soon as TD transfer has completed.

This field is only used if Variable Peripheral Select is active (PS = 1).

Figure 40-30.Clock Synchronization in Write Mode


TWCK
CLOCK is tied low by the TWI as long as RHR is full
TWD
S
SADR
W
A
DATA0
A
DATA1
A
DATA2
NA
S
ADR
TWI_RHR
DATA0 is not read in the RHR
DATA1
DATA2
SCLWS
SCL is stretched on the last bit of DATA1
RXRDY
Rd DATA0
Rd DATA1
Rd DATA2
SVACC
SVREAD
TXCOMP
As soon as a START is detected

Notes: 1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.

2. SCLWS is automatically set when the clock synchronization mechanism is started and automatically reset when the mechanism is finished.

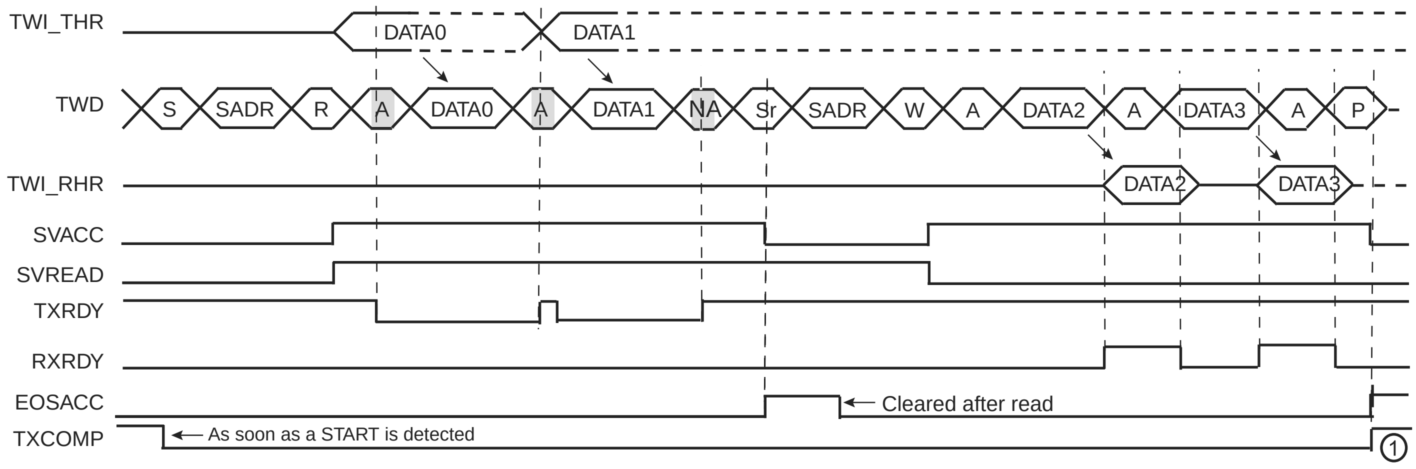
40.10.5.5 Reversal after a Repeated Start

Reversal of Read to Write

The master initiates the communication by a read command and finishes it by a write command.

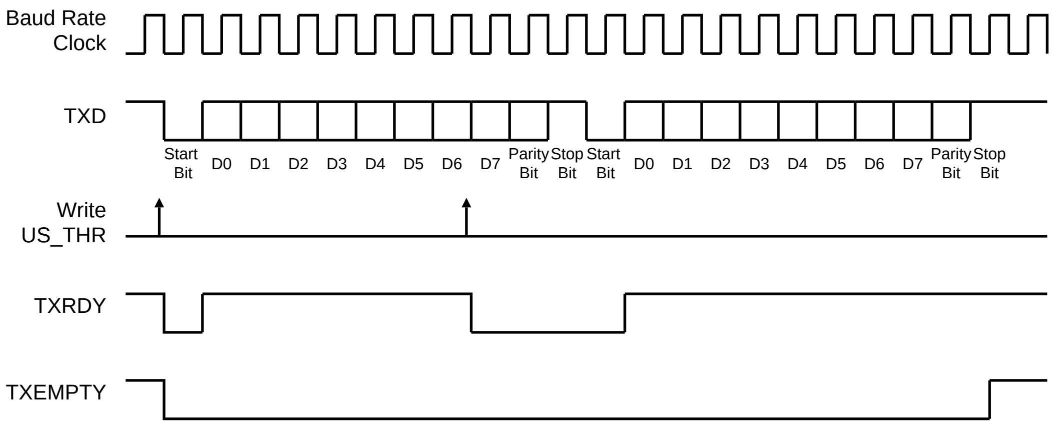
Figure 40-31 describes the repeated start + reversal from Read to Write mode.

Figure 40-31.Repeated Start + Reversal from Read to Write Mode



Note: 1. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

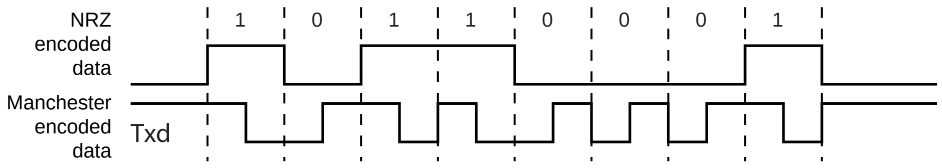
Figure 44-7. Transmitter Status



44.7.3.2 Manchester Encoder

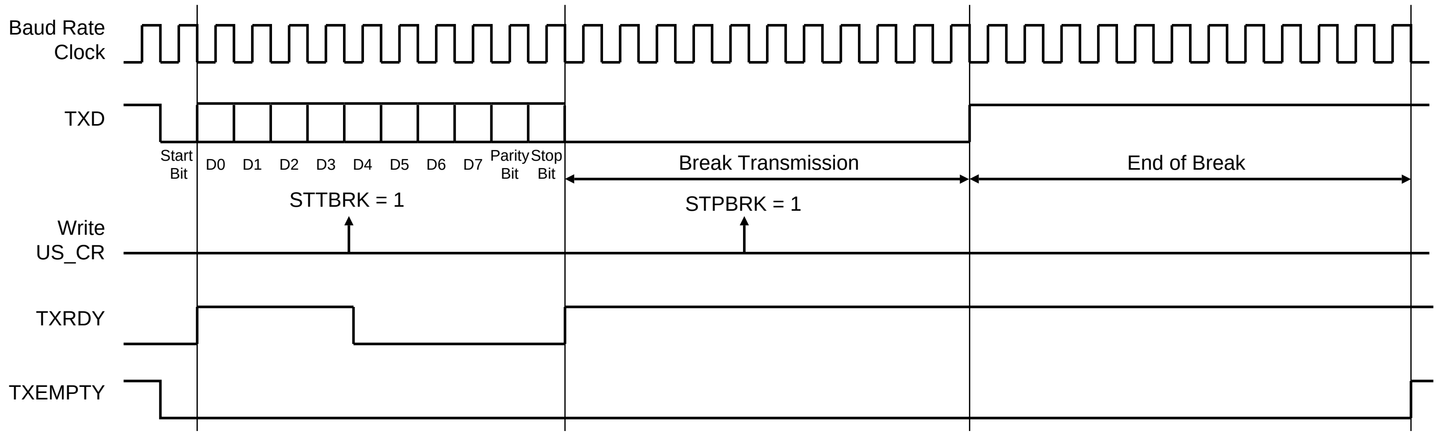
When the Manchester encoder is in use, characters transmitted through the USART are encoded based on biphase Manchester II format. To enable this mode, set the MAN field in the US_MR register to 1. Depending on polarity configuration, a logic level (zero or one), is transmitted as a coded signal one-to-zero or zero-to-one. Thus, a transition always occurs at the midpoint of each bit time. It consumes more bandwidth than the original NRZ signal (2x) but the receiver has more error control since the expected input must show a change at the center of a bit cell. An example of Manchester encoded sequence is: the byte 0xB1 or 10110001 encodes to 10 01 10 10 01 01 01 10, assuming the default polarity of the encoder. Figure 44-8 illustrates this coding scheme.

Figure 44-8. NRZ to Manchester Encoding



The Manchester encoded character can also be encapsulated by adding both a configurable preamble and a start frame delimiter pattern. Depending on the configuration, the preamble is a training sequence, composed of a predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to 0, the preamble waveform is not generated prior to any character. The preamble pattern is chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE, writing the field TX_PP in the US_MAN register, the field TX_PL is used to configure the preamble length. Figure 44-9 illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using the TX_MPOL field in the US_MAN register. If the TX_MPOL field is set to zero (default), a logic zero is encoded with a zero-to-one transition and a logic one is encoded with a one-to-zero transition. If the TX_MPOL field is set to one, a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

Figure 44-26.Break Transmission



44.7.3.14 Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. This corresponds to detecting a framing error with data to 0x00, but FRAME remains low.

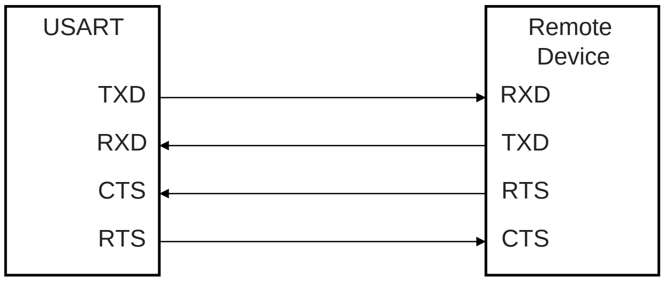
When the low stop bit is detected, the receiver asserts the RXBRK bit in US_CSR. This bit may be cleared by writing the Control Register (US_CR) with the bit RSTSTA to 1.

An end of receive break is detected by a high level for at least 2/16 of a bit period in asynchronous operating mode or one sample at high level in synchronous operating mode. The end of break detection also asserts the RXBRK bit.

44.7.3.15 Hardware Handshaking

The USART features a hardware handshaking out-of-band flow control. The RTS and CTS pins are used to connect with the remote device, as shown in Figure 44-27.

Figure 44-27.Connection with a Remote Device for Hardware Handshaking



Setting the USART to operate with hardware handshaking is performed by writing the USART_MODE field in the Mode Register (US_MR) to the value 0x2.

The USART behavior when hardware handshaking is enabled is the same as the behavior in standard synchronous or asynchronous mode, except that the receiver drives the RTS pin as described below and the level on the CTS pin modifies the behavior of the transmitter as described below. Using this mode requires using the DMAC channel for reception. The transmitter can handle hardware handshaking in any case.

Figure 44-28 shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processing, the transmitter is disabled only after the completion of the current character and transmission of the next character happens as soon as the pin CTS falls.

44.8.20 USART IrDA FILTER Register

Name: US_IF

Address: 0xF001C04C (0), 0xF002004C (1), 0xF802004C (2), 0xF802404C (3)

Access: Read-write

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
IRDA_FILTER							

This register is relevant only if USART_MODE = 0x8 in “USART Mode Register” on page 1407.

This register can only be written if the WPEN bit is cleared in “USART Write Protect Mode Register” on page 1431.

- **IRDA_FILTER: IrDA Filter**

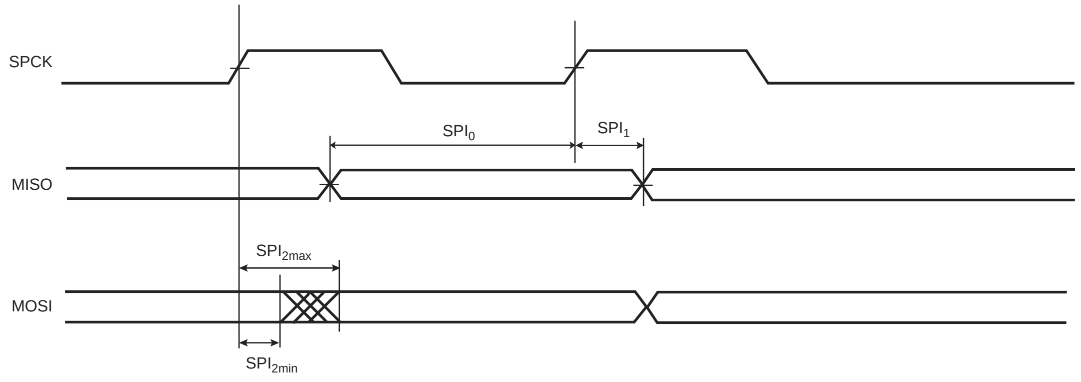
The IRDA_FILTER value must be defined to meet the following criteria:

t_{MCK} * (IRDA_FILTER + 3) < 1.41 µs

Table 54-44. SPI Timings with 1v8 Peripheral Supply (Continued)

Symbol	Parameter	Conditions	Min	Max	Unit
SPI_8	MOSI Hold time after SPCK rises	—	1	—	ns
SPI_9	SPCK rising to MISO	—	2.9(1)	10.9(1)	ns
SPI_{10}	MOSI Setup time before SPCK falls	—	1.7	—	ns
SPI_{11}	MOSI Hold time after SPCK falls	—	1	—	ns
SPI_{12}	NPCS0 setup to SPCK rising	—	2.1	—	ns
SPI_{13}	NPCS0 hold after SPCK falling	—	19	—	ns
SPI_{14}	NPCS0 setup to SPCK falling	—	3.2	—	ns
SPI_{15}	NPCS0 hold after SPCK rising	—	19.6	—	ns
SPI_{16}	NPCS0 falling to MISO valid	—	—	10.6	ns

Figure 54-14.Min and Max access time for SPI output signal


SPCK
MISO
MOSI
SPI_0
SPI_1
SPI_{2max}
SPI_{2min}