AN13028
Advanced HyperRAM/PSRAM Usage on i.MX RT

Rev. 0 — 10/2020

Contents

1 Introduction

The i.MX RT series MCU is a crossover product from NXP. It includes a FlexSPI controller which supports HyperBus devices(HyperFlash/HyperRAM), Serial NOR Flash or other device with similar SPI protocol as Serial NOR Flash, etc.

How to Enable HyperRAM with i.MX RT (document AN12239) introduces the basic knowledge about how to enable HyperRAM with i.MX RT. This application note describes the advanced usage of HyperRAM/PSRAM when used with FlexSPI on i.MX RT MCU, including FlexSPI prefetch function, HyperRAM/PSRAM refresh interval, and HyperRAM devices supported for i.MX RT.

In this application note, i.MX RT1050 and RT1170 are used as examples, and HyperRAM devices from different venders and PSRAM device from Apmemory are used for testing.

2 FlexSPI controller

Flexible Serial Peripheral Interface (FlexSPI) controller in i.MX RT1050 supports following features:

- Flexible sequence engine (LUT table) to support various vendor devices:
    - Serial NOR/NAND Flash
    - HyperBus device (HyperFlash/HyperRAM)
    - FPGA device
- Flash access mode:
    - Single/Dual/Quad/Octal mode
    - SDR/DDR mode
    - Individual/Parallel mode
- Memory mapped read/write access by AHB bus:
    - AHB RX buffer implemented to reduce read latency. Total AHB RX buffer size: 128 × 64 bits
    - Four flexible and configurable buffers in AHB RX buffer
    - AHB TX buffer implemented to buffer all write data from one AHB burst. AHB TX buffer size: 8 × 64 bits
- Software triggered Flash read/write access by IP bus:
    - IP RX FIFO implemented to buffer all read data from the external device. FIFO size: 16 × 64 bits
    - IP TX FIFO implemented to buffer all Write data to the external device. FIFO size: 16 × 64 bits

3 HyperBus interface

FlexSPI controller can support HyperBus device, and HyperRAM is the device used with HyperBus interface.

HyperBus is a low signal count, Double Data Rate (DDR) interface, that achieves high-speed read and write throughput. Command, address, and data information is transferred over the eight HyperBus DQ[7:0] signals. The clock (CK#, CK) is used for information capture by a HyperBus slave device when receiving command, address, or data on the DQ signals. Command or Address values are center-aligned with clock transitions.

Every transaction begins with the assertion of CS# and Command-Address (CA) signals, followed by the start of clock transitions to transfer six CA bytes, followed by initial access latency and either read or write data transfers, until CS# is deasserted.

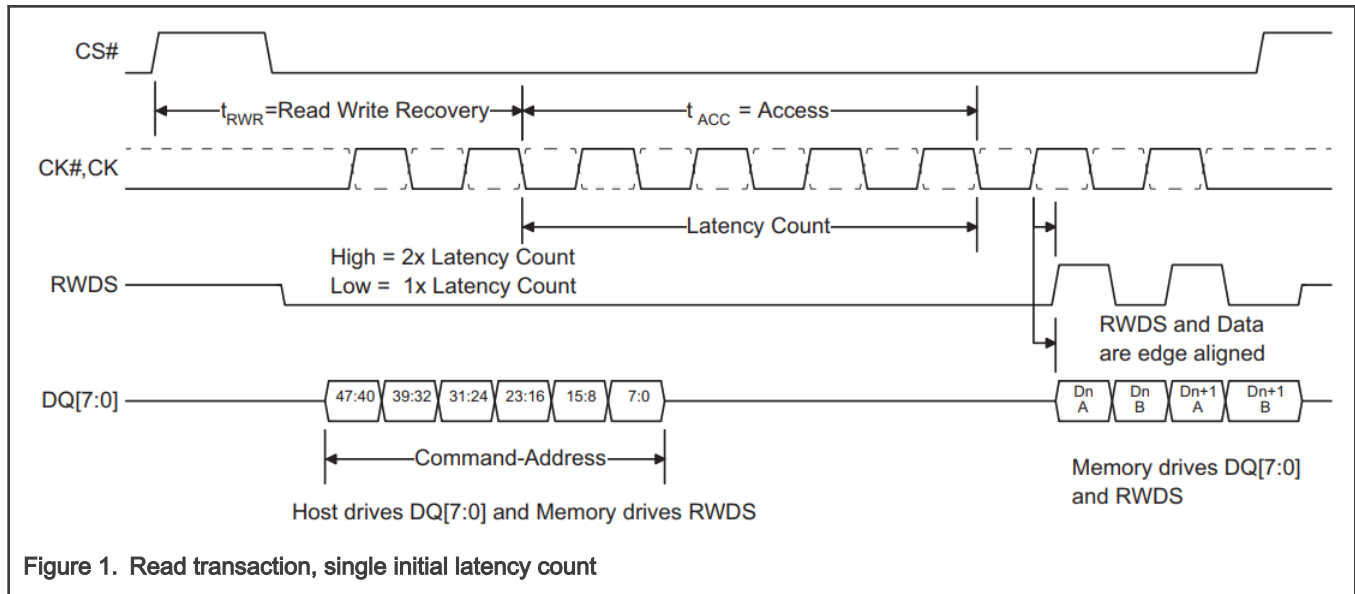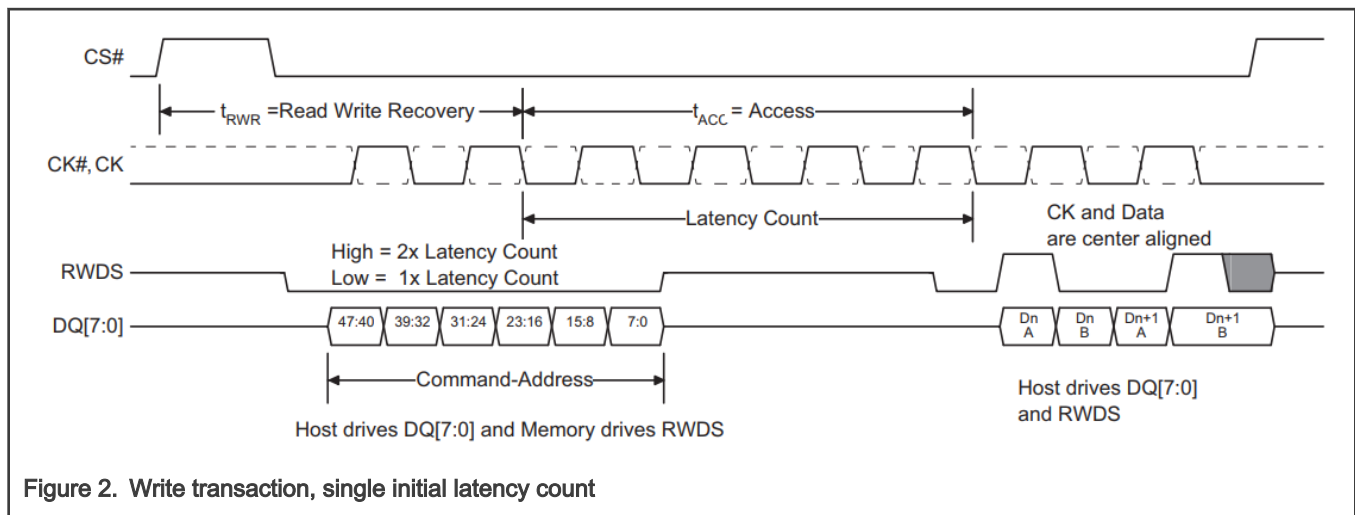Figure 1 shows a read transaction with single latency count.

Figure 1. Read transaction, single initial latency count

Figure 2 shows a write transaction with single latency count.

Figure 2. Write transaction, single initial latency count

4 PSRAM overview

This chapter briefly describes the features of PSRAM devices used in this application note.

4.1 QSPI PSRAM

One PSRAM device used is APS6404L-3SQR. It is the QSPI PSRAM from apmemory vendor.

This PSRAM device features a high speed, low pin count interface. It has four Single Data Rate (SDR) I/O pins and operates in Serial Peripheral Interface (SPI) or Quad Peripheral Interface (QPI) mode with frequencies up to 133 MHz. The data input (A/DQ) to the memory relies on clock (CLK) to latch all instructions, addressesand data. It is most suitable for low-power and low-cost portable applications. It incorporates a seamlessself-managed refresh mechanism. Hence it does not require the support of DRAM refresh from systemhost.

SPI/QPI PSRAM device is byte-addressable. The device recognizes the following commands specified by the various input methods.

Command	Code	SPI Mode (QE=0)					QPI Mode (QE=1)				
		Cmd	Addr	Wait Cycle	DIO	Max Freq.	Cmd	Addr	Wait Cycle	DIO	Max Freq.
Read	'h03	S	S	0	S	33	N/A				
Fast Read	'h0B	S	S	8	S	133/84*	Q	Q	4	Q	66
Fast Read Quad	'hEB	S	Q	6	Q	133/84*	Q	Q	6	Q	133/84*
Write	'h02	S	S	0	S	133/84*	Q	Q	0	Q	133/84*
Quad Write	'h38	S	Q	0	Q	133/84*	same as 'h02				
Enter Quad Mode	'h35	S	-	-	-	133	N/A				
Exit Quad Mode	'hF5	N/A					Q	-	-	-	133
Reset Enable	'h66	S	-	-	-	133	Q	-	-	-	133
Reset	'h99	S	-	-	-	133	Q	-	-	-	133
Wrap Boundary Toggle	'hC0	S	-	-	-	133	Q	-	-	-	133
Read ID	'h9F	S	S	0	S	33	N/A				
Remark: S = Serial IO, Q = Quad IO											

Figure 3. Command/Address latching truth table

4.2 Octal SPI PSRAM

Another PSRAM device used is APS12808L-OBM-BA. It is the Octal SPI PSRAM from apmemory vendor.

This PSRAM device has eight Double Data Rate (DDR) I/O pins which transfers two bytes per one clock cycle andoperates in SPI mode with frequencies up to 200 MHz.

Octal DDR PSRAM device is also byte-addressable. However, Memory accesses are required to start on even addresses (A[0]='0') and Mode Register accesses allow both even and odd addresses.

The Octal DDR PSRAM recognizes the following commands specified on the INST (Instruction) cycle defined by the Address/DQ pins.

Command	1st CLK		2nd CLK		3rd CLK	
Sync Read	00h		A3	A2	A1	A0
Sync Write	80h		A3	A2	A1	A0
Sync Read (Linear Burst)	20h		A3	A2	A1	A0
Sync Write (Linear Burst)	A0h		A3	A2	A1	A0
Mode Register Read	40h		×			MA
Mode Register Write	C0h		×			MA
Global Reset	FFh		×			

Remarks:		
	×	= don't care (V_{IH}/V_{IL})
	A3	= unused address bits are reserved
	A2	= RA[13:6]
	A1	= RA[5:0],CA[9:8]
	A0	= CA[7:0]
	MA	= Mode Register Address

Figure 4. Command truth table

5 Advanced usage

This chapter describes advanced knowledges when using HyperRAM/PSRAM with i.MX RT series. It includes HyperRAM/PSRAM devices validated and some special configurations for FlexSPI and HyperRAM/PSRAM.

5.1 Prefetch

FlexSPI supports AHB RX prefetch buffer, and it is used for prefetching data when reading the external memory (for example, HyperRAM) to improve access latency during the read access and overall performance.

There are four buffers in AHB RX buffer for i.MX RT1050. AHB RX buffer's total size is 1 Kbytes. The buffer size is flexibly configurable for each buffer in AHB RX buffer by register fields `AHBRXBUF0CR0[BUFSZ]` - `AHBRXBUF3CR0[BUFSZ]`.

Figure 5. AHBRXBUF0CR0 register
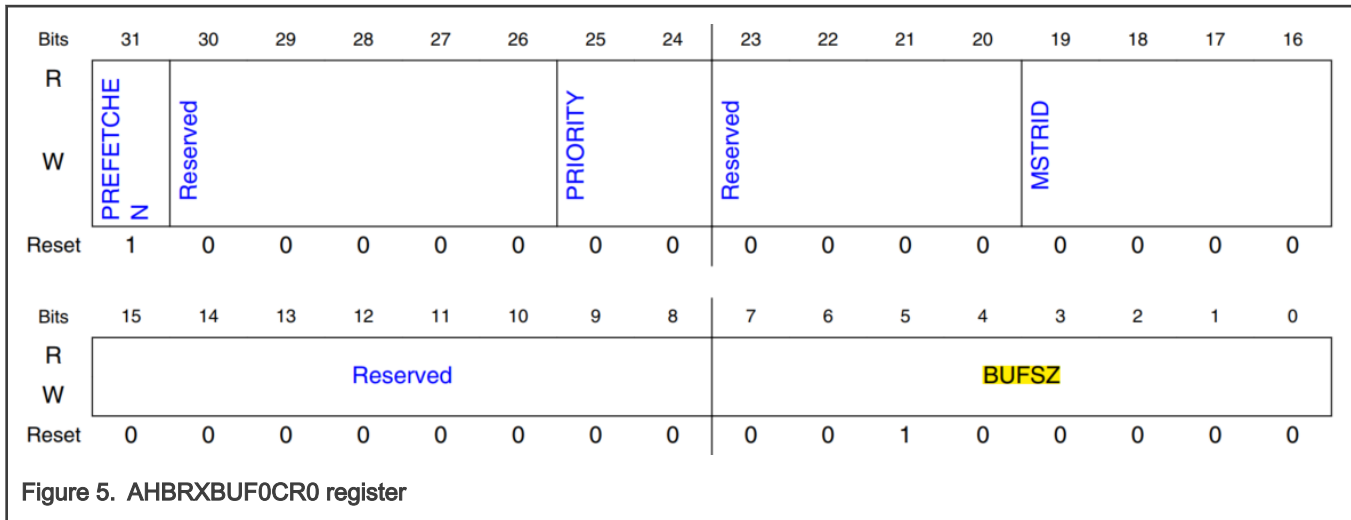
FlexSPI specifies different buffer size for different master. That means some masters may have the dedicated prefetch buffer, which can optimize performance in some applications. The AHB RX buffer is assigned to different masters according to the register fields, `AHBRXBUF0CR0[MSTRID]` - `AHBRXBUF3CR0[MSTRID]`.

Some masters are assigned with dedicated master ID, as shown in Table 1. The independent master ID is assigned to core, eDMA and DCP. Other masters share one ID, say PXP, USB and so on.

Table 1. Master IDs

Module	Master ID
Core platform	000b
eDMA	001b
DCP	010b
All others	011b

AHB read prefetch is enabled or disabled by modifying register fields, `AHBCR[PREFETCHEN]` and `AHBRXBUFxCR0[PREFETCHEN]`. AHB prefetch is enabled only when both when both `AHBCR[PREFETCHEN]=0x1` and `AHBRXBUFxCR0[PREFETCHEN]=0x1` (x is the AHB RX buffer ID for current AHB master). The prefetch data size is determined by AHB buffer size.

Figure 6. AHBCR register
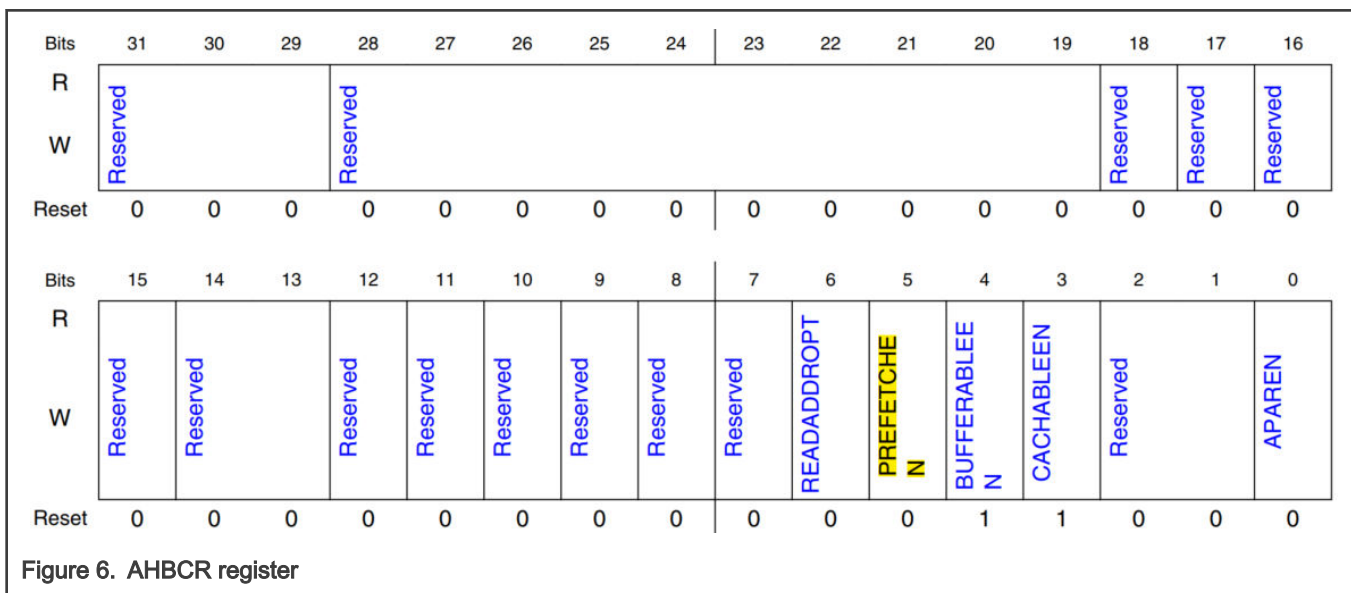
The following describes the specific process of AHB prefetch, taking access to HyperRAM as an example.

When AHB prefetch is enabled, once receiving the request from the bus, it first checks whether the request matched the current AHB buffer address range. If yes, it directly returns the data. If not, it triggers to access HyperRAM to read new data to AHB buffer. After returning the required data to bus, it continues to prefetch data in following address to AHB buffer until the buffer is full. AHB read prefetch can be aborted by any new IP command or AHB command request when it is prefetching data.

Figure 7 shows the general read prefetching flow.
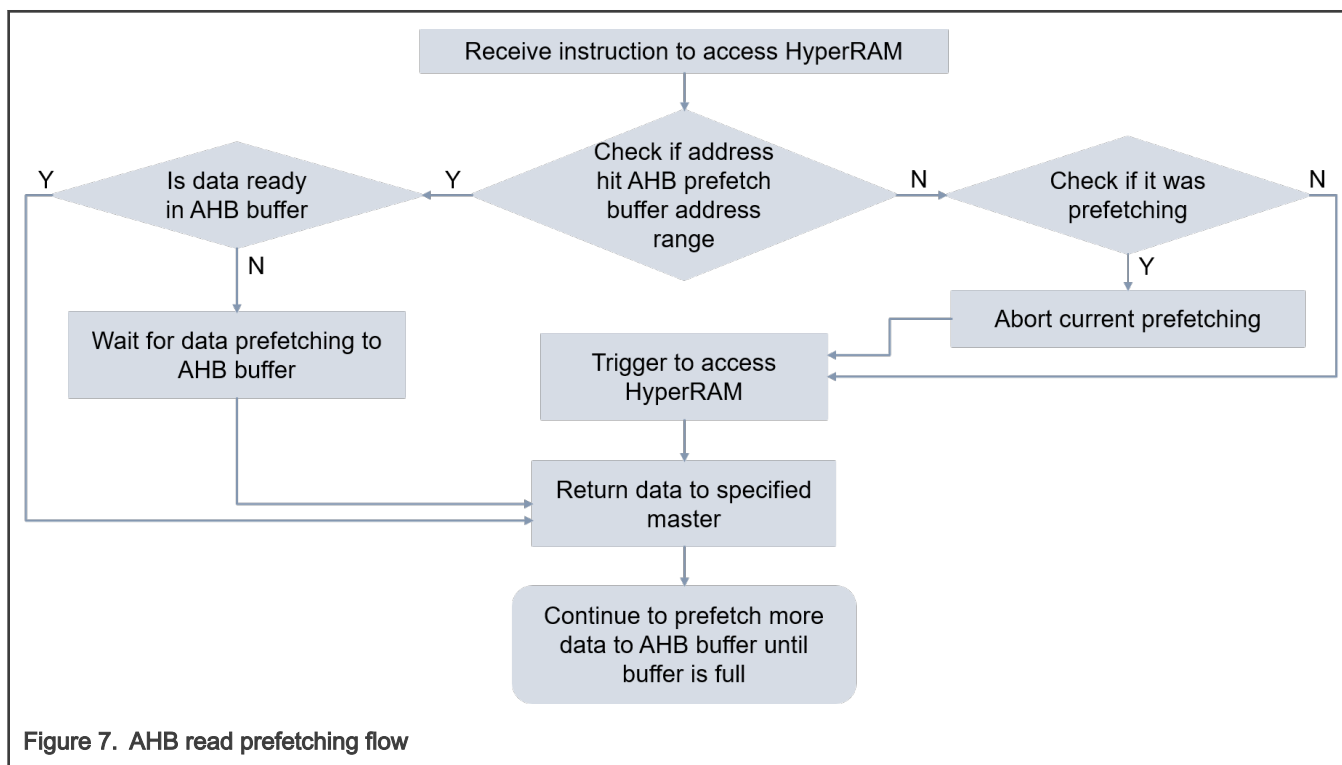
Figure 7. AHB read prefetching flow

5.2 TCSH/TCSS fields

FlexSPI should follow specific timing between Chip Selection (CS, on `A_SS0_B`/`A_SS1_B`/`B_SS0_B`/`B_SS1_B`) and SCLK signals.

For SDR mode, the delay from chip select assertion and the SCLK rising edge is (TCSS+0.5) cycles of serial root clock. The delay from SCLK falling edge and chip select de-assertion is TCSH cycles of serial root clock. Figure 8 indicates the timing relationship between chip selection and SCLK.
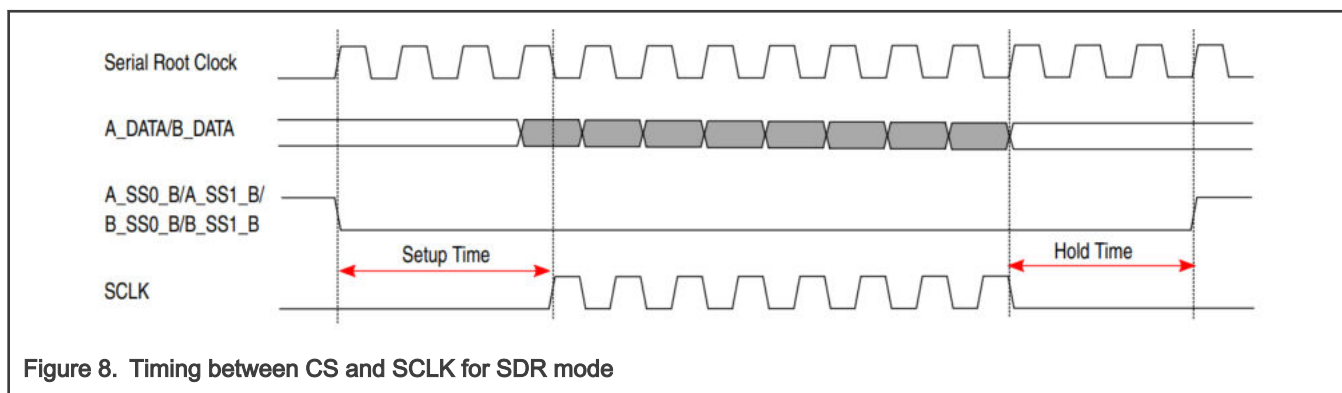
Figure 8. Timing between CS and SCLK for SDR mode

For DDR mode, the delay from chip selection assertion and SCLK rise edge is (TCSS+0.5) cycles of serial root clock. The delay from SCLK fall edge and chip selection deassertion is (TCSH+0.5) cycles of serial root clock. Figure 9 indicates the timing relationship between chip selection and SCLK.
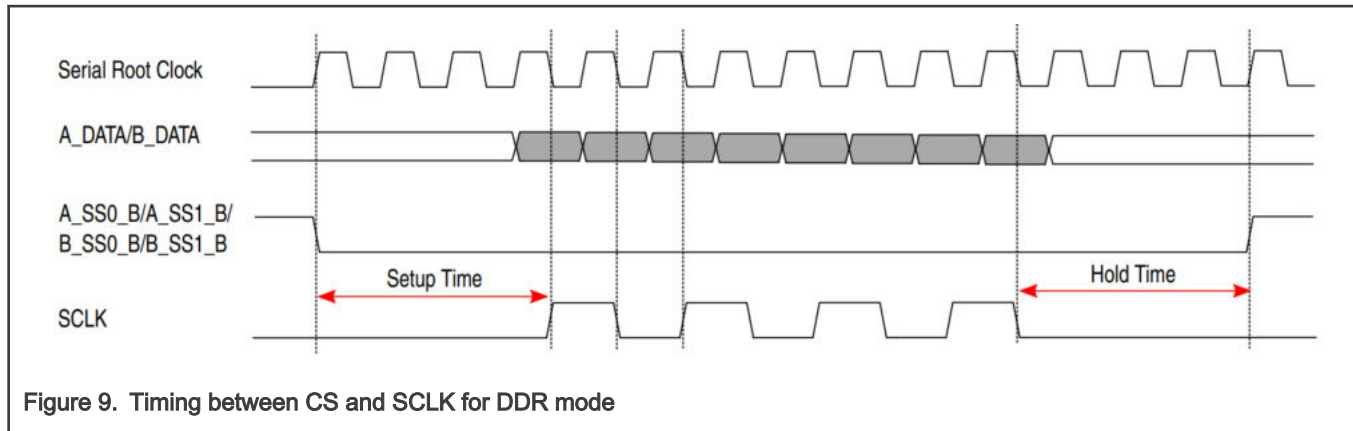
Figure 9. Timing between CS and SCLK for DDR mode

HyperRAM is the device using HyperBus interface and HyperBus interface supports DDR mode. Figure 9 shows the configuration of delay between CS and SCLK. APS6404L PSRAM supports SDR mode, while APS12808L PSRAM supports DDR mode.

TCSH and TCSS are fields of `FLSHAxCR1`/`FLSHBxCR1` (x = 1,2) registers. Figure 10 shows the diagram of `FLSHAxCR1`/`FLSHBxCR1` registers.
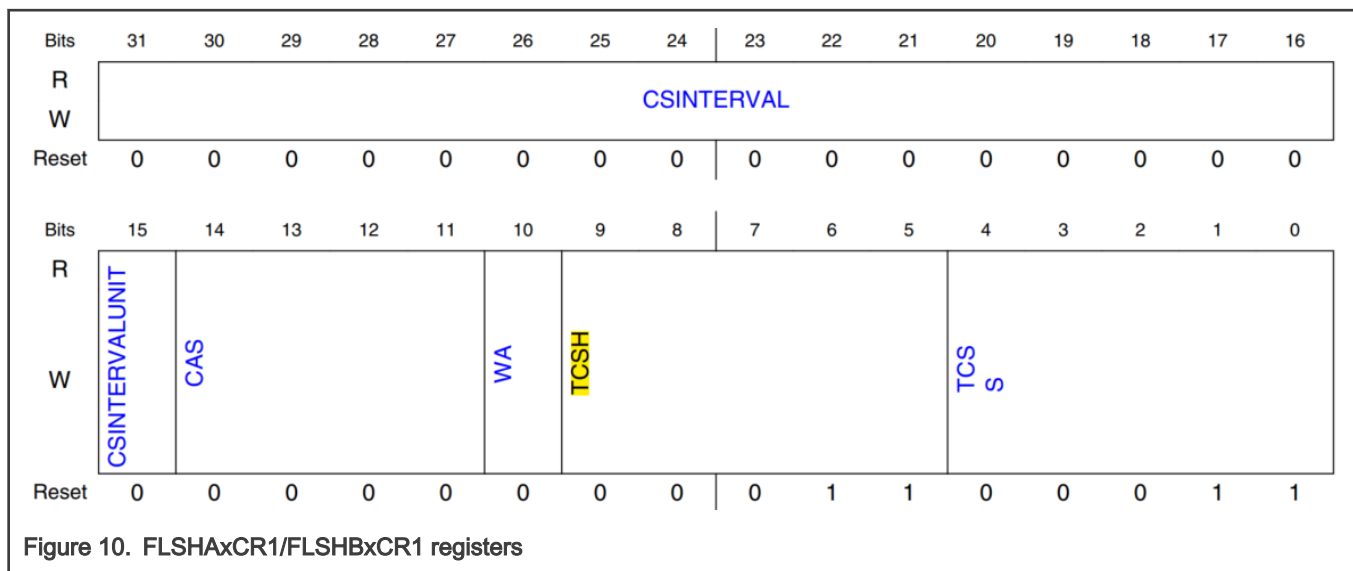

Figure 10. FLSHAxCR1/FLSHBxCR1 registers

For example, for HyperRAM device, TCSH and TCSS is equivalent to the t_{CSH} and t_{CSS} timing parameters. For example, Table 2 shows the related parameters in 7KS0642GAHI02 HyperRAM device.

Table 2. HyperRAM timing parameters

Parameter	Symbol	200 MHz		166 MHz		Unit
		Min.	Max.	Min.	Max.	
Chip select setup to next CK rising Edge	t_{CSS}	4.0	—	3	—	ns
Chip select hold after CK failing Edge	t_{CSH}	0	—	0	—	ns

In general, TCSH and TCSS values can be configured according to the timing parameters of HyperRAM, as shown in Table 2. However, as shown in Figure 11, if TCSH is set to **0**, when prefetch is aborted (prefetch is enabled), the SCLK might be reset to return to default level after CS signal is deasserted, which may result in read errors.

Figure 11. TCSH set as 0
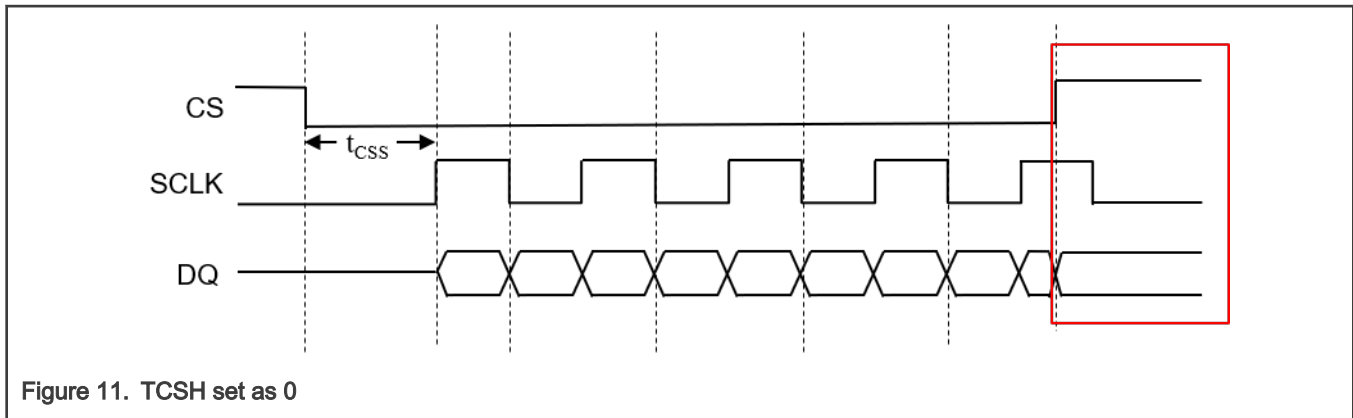
Therefore, it is recommended to set TCSH and TCSS to the default values. Set both TCSH and TCSS to **3**, as shown in Figure 10.

5.3 Distributed refresh interval

The DRAM array requires periodic refresh of all bits in the array. This can be done by the host system by reading or writing a location in each row within a specified time limit. The read or write access copies a row of bits to an internal buffer. At the end of the access the bits in the buffer are written back to the row in memory, thereby recharging (refreshing) the bits in the row of DRAM memory cells.

HyperRAM devices include self-refresh logic that will refresh rows automatically. The automatic refresh of a row can only be done when the memory is not being actively read or written by the host system. The refresh logic waits for the end of any active read or write before doing a refresh, if a refresh is needed at that time. If a new read or write begins before the refresh is completed, the memory will drive RWDS HIGH during the CA period to indicate that an additional initial latency time is required at the start of the new access in order to allow the refresh operation to complete before starting the new access.

Table 3 shows the required refresh interval for the entire memory array varies with temperature of Cypress HyperRAM. This is the time within which all rows must be refreshed. Refresh of all rows could be done as a single batch of accesses at the beginning of each interval, in groups (burst refresh) of several rows at a time, spread throughout each interval, or as single row refreshes evenly distributed throughout the interval. The self-refresh logic distributes single row refresh operations throughout the interval so that the memory is not busy doing a burst of refresh operations for a long period, such that the burst refresh would delay host access for a long period.

Table 3. Array refresh interval per temperature

Device temperature (°C)	Array refresh interval (ms)	Array rows	Recommended t_{CSM} (µs)
85	64	8192	4
105	16	8192	1

The distributed refresh method requires that the host does not do burst transactions that are so long as to prevent the memory from doing the distributed refreshes when they are needed. This sets an upper limit on the length of read and write transactions so that the refresh logic can insert a refresh between transactions. This limit is called the CS# LOW maximum time (t_{CSM}). The t_{CSM} value is determined by the array refresh interval divided by the number of rows in the array, then reducing this calculation by half to ensure that a distributed refresh interval cannot be entirely missed by a maximum length host access starting immediately before a distributed refresh is needed.

When using HyperRAM, the host system is required to respect the t_{CSM} value by ending each transaction before violating t_{CSM}. This can be done by host memory controller logic splitting long transactions when reaching the t_{CSM} limit, or by host system hardware or software not performing a single read or write transaction that would be longer than t_{CSM}. Otherwise, it will cause HyperRAM to fail to do self-refresh, which will lead to unpredictable errors.

In order to avoid violating t_{CSM}, it is recommended to follow the below suggestions:

- AHB RX buffer size, as defined by BUFSZ field in `AHBRXBUFxCR0` register, should not be too large. When prefetch is enabled, bigger AHB RX buffer size will make a single read access operation take too long time, which may violate t_{CSM}.

- Clock for FlexSPI and AHB Bus should not be too small. Smaller clock frequencies make a single access operation take longer time, which may violate t_{CSM}.

5.4 Validated HyperRAM devices

HyperRAM devices from different vendors are tested to validate if they can work well with i.MX RT series. Table 4 lists the test results of all HyperRAM devices.

As shown in Table 4, Cypress HyperRAM device, **7KS0641DPHI02**, seems to have an issue which will enter a hard fault during testing when enabling read prefetch of FlexSPI.

The cause of this issue is that the **7KS0641DPHI02** device seems to have an issue when there is a prefetch abort in the middle of command/address cycle. So prefetch abort during command/address cycle should be prevented to avoid the issue, which is hard to implement for FlexSPI. Therefore, it is not recommended to use **7KS0641DPHI02** HyperRAM on i.MX RT series.

When using HyperRAM on i.MX RT series, all passed devices in Table 4 are recommended to use.

Table 4. HyperRAM devices test results

RT part number	HyperRAM vendor	HyperRAM part number	Results
PIMXRT1176DVMAA	Cypress	7KL0642DPHB02 (8 MB)	PASS
PIMXRT1064DVL6A	Cypress	7KS0642GAHI02 (8 MB)	PASS
PIMXRT1052DVL6B	Cypress	7KS0641DPHI02 (8 MB)	FAIL
PIMXRT1064DVL6A	Cypress	7KS0641DPHI02 (8 MB)	FAIL
PIMXRT1064DVL6A	Winbond	W956x8MBYA (8 MB)	PASS
PIMXRT1064DVL6A	ISSI	IS66WVH32M8DALL (32 MB)	PASS

5.5 PSRAM test

This section describes PSRAM devices test on i.MX RT series and summarizes the PSRAM devices currently verified on RT series.

5.5.1 QSPI PSRAM

For QSPI PSRAM, all i.MX RT series can support well.

One PSRAM device APS6404L-3SQR from apmemory vendor has been tested on i.MX RT1050. The test result is that this PSRAM can work well with i.MX RT1050.

5.5.2 Octal SPI PSRAM

For Octal SPI PSRAM, i.MX RT1020/RT1050/RT1060 have some limitation to support.

As mentioned above, memory accesses of Octal SPI PSRAM are required to start on even addresses (A[0]='0). However, FlexSPI controller of RT1020/RT1050/RT1060 has one limitation that can't support this access method of Octal SPI PSRAM.

Fortunately, after making specific updates for FlexSPI IP, now RT1010 and RT1170 products can support Octal SPI PSRAM. In order to confirm this, PSRAM device APS12808L-OBM-BA is used to test on i.MX RT1170.

In order to support Octal SPI PSRAM, two points should be paid attention to in FlexSPI configuration:

- For AHB read from PSRAM, READADDROP bit of AHBCR register, as shown in Figure 12, must be set to **1**. This will remove AHB burst start address alignment limitation.

- For AHB write to PSRAM, WMOPT1 bit of FLSHCR4 register, as shown in Figure 12, must be set to **0**. This will remove AHB write burst minimal length limitation for Octal SPI PSRAM.

Figure 12. FLSHCR4 register
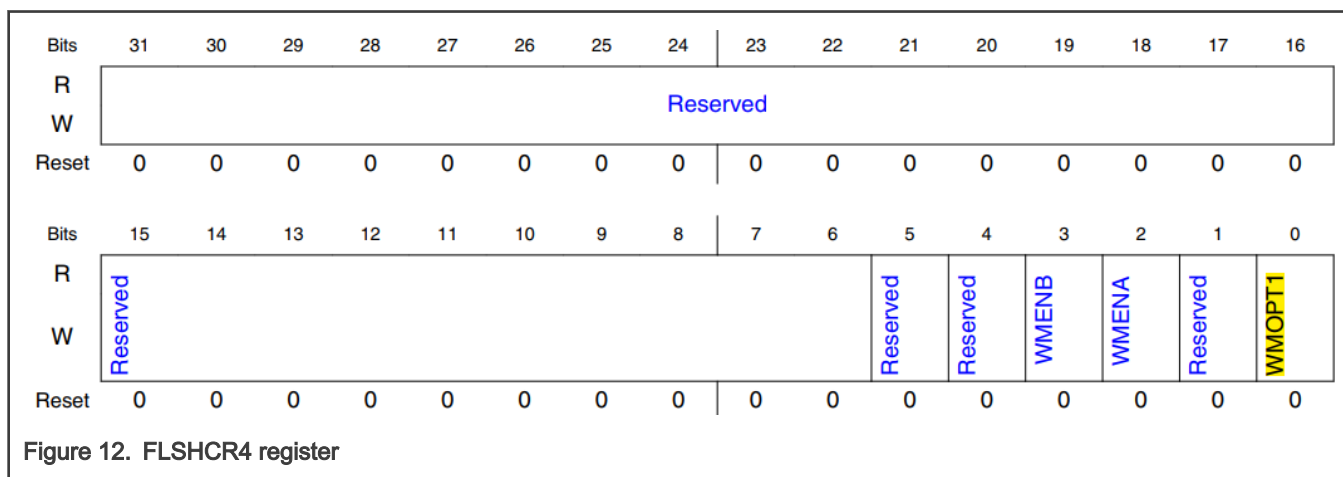
The test result shows that RT1170 can support Octal SPI PSRAM well.

5.5.3 Supported PSRAM

Table 5 lists validated and supported PSRAM devices on i.MX RT series. For QSPI PSRAM devices, all RT series can support well, while for Octal SPI PSRAM, only RT1010/RT1170 and all RTyyy products can support well.

Table 5. PSRAM support table

PSRAM interface	PSRAM vendor	PSRAM part number	RT product	Support or not
QSPI	Apmemory	APS6404L-3SQR	RT1050	Yes
	—	—	All RTxxxx	Yes
			All RTyyy	Yes
Octal SPI	Apmemory	APS12808L-OBM-BA	RT1170	Yes
	—	—	RT1010	Yes
			All RTyyy	Yes
			RT1020/RT1050/ RT1060	No

6 References

1. *i.MX RT1050 Processor Reference Manual* (document MXRT1050RM)

2. *How to Enable HyperRAM with i.MX RT* (document AN12239)

arm