**QSPI**

From the QSPI controller specification:

- The QSPI-FSM handles serialization/deserialization, clock generation, and pin control.

- It receives command opcode and flash address from the Command Engine (CE) or XIP Engine.

- It uses TX and RX FIFOs for data.

The Macronix flash (MX25L6436F) supports:

- Various read commands: READ (03h), FAST\_READ (0Bh), 2READ (BBh), DREAD (3Bh), 4READ (EBh), QREAD (6Bh).

- For quad I/O, we are particularly interested in 4READ (EBh) and QREAD (6Bh).

The QSPI controller specification allows configuring the number of lanes for each phase (command, address, data, dummy) via the CMD\_CFG or XIP\_CFG registers.

**States:**

* **IDLE**
* **CMD :** Send command opcode (number of lanes = CMD\_LANES)
* **ADDR :** Send address (number of lanes = ADDR\_LANES, number of bytes = ADDR\_BYTES)
* **MODE :** Send mode bits (if MODE\_EN is set)
* **DUMMY :** Dummy cycles (number of cycles = DUMMY\_CYCLES + EXTRA\_DUMMY)
* **DATA\_TX :** Transmit data (for write operations, using DATA\_LANES)
* **DATA\_RX :** Receive data (for read operations, using DATA\_LANES)
* **STOP\_CS :** Deassert CS#

**Handle the dummy cycles in the state machine**

- The dummy state will last for the number of cycles (each cycle is one SPI clock, and since in quad mode, one cycle sends 4 bits? Actually, during dummy cycles, no data is driven by the controller or the flash? The flash expects the dummy cycles to be cycles where the clock is toggled and the lines are in a high-impedance state? The controller doesn't drive data during dummy cycles for read operations. For write operations, the dummy cycles might be used for the flash to prepare, but in read the dummy cycles are for the flash to prepare the data.

Therefore, in the DUMMY state, just count the clock cycles and do nothing on the data lines.

Note that in the quad mode, the IO lines are bidirectional. During dummy cycles for a read command, the controller should set the IO pins as inputs (or high-Z) because the flash might require the lines to be in a known state (like driven high by pull-ups) or might not care.

**Implementation:**

In the DUMMY state, the controller will:

- For each dummy cycle, generate one clock cycle and keep the IO lines as inputs (for read).

Since the dummy cycles are part of the read command, the controller should not drive the IO lines during dummy cycles.

The state machine will count the dummy cycles and then transition to the data phase.

For the data phase:

- For read: the controller will capture data on the falling (or rising, depending on CPHA) edge of the clock on the configured number of lanes (4 for quad).

- For write: the controller will drive data on the lanes.

**State Machine States:**

**IDLE:**

- Wait for a start signal (from CE or XIP engine).

- When started, assert CS# and go to CMD state.

CMD:

- Shift out the command opcode (from the CMD\_OP register) on the number of lanes specified by CMD\_LANES.

- The number of bits is 8, so the number of clock cycles = ceil(8 / CMD\_LANES). For single: 8 cycles, dual: 4 cycles, quad: 2 cycles.

- After sending the command, go to ADDR state (if ADDR\_BYTES > 0) or MODE state (if MODE\_EN) or DUMMY state (if dummy cycles>0) or DATA state (if data phase) or STOP\_CS (if no more phases).

**ADDR:**

- Shift out the address (from CMD\_ADDR register) on ADDR\_LANES.

- The number of bits = ADDR\_BYTES \* 8, so clock cycles =

ceil((ADDR\_BYTES\*8) / ADDR\_LANES).

- Then go to MODE state if MODE\_EN is set, else to DUMMY state if dummy cycles>0, else to DATA state (if data phase) or STOP\_CS.

**MODE:**

- Shift out the mode bits (from CMD\_OP[15:8] or XIP\_CMD[23:16]) on the same lanes as the address

- The number of mode bits is 8? (The IP spec has 8 bits in the registers). So clock cycles = ceil(8 / MODE\_LANES). But the IP spec doesn't have a separate MODE\_LANES. We assume the same as ADDR\_LANES.

- Then go to DUMMY state if dummy cycles>0, else to DATA state or STOP\_CS.

**DUMMY:**

- Cycle the clock for the number of dummy cycles (each cycle is one clock pulse, but note: in quad mode, one cycle is one clock and all 4 lanes are in high-Z).

- The controller does not drive data during dummy cycles for read commands. For write commands, dummy cycles are not typically used.

- After the dummy cycles, go to DATA state (if data phase) or STOP\_CS.

**DATA:**

- If it's a read (DIR=1), then capture data from the lanes (DATA\_LANES) for the number of bytes in CMD\_LEN (or until CS# is deasserted in XIP mode).

- If it's a write (DIR=0), then send data from the TX FIFO on the lanes (DATA\_LANES) for the number of bytes in CMD\_LEN.

- The state machine must handle the entire data transfer (which might be long and involve FIFO).

**STOP\_CS:**

- Deassert CS# and then go back to IDLE.

Additionally, we must handle the clock polarity (CPOL) and phase (CPHA) as set in the CTRL register.

**Key Features**:

* Supports **1/2/4 data lanes** (Single/Dual/Quad SPI).
* Configurable **address length** (0/3/4 bytes), **dummy cycles** (0–15), and **mode bits**.
* Bidirectional I/O control for Quad I/O operations.
* Implements **SPI modes 0 & 3** (CPOL/CPHA).

**2. State Machine States**

| **State** | **Description** | **Operations** |
| --- | --- | --- |
| **IDLE** | Wait for transaction | Assert CS# on start signal. |
| **CMD** | Send opcode | Shift 8-bit command (1/2/4 lanes). LSB/MSB first per LSB\_FIRST. |
| **ADDR** | Send address | Shift 24/32-bit address (1/2/4 lanes). |
| **MODE** | Send mode bits | Shift 8-bit mode data (if MODE\_EN=1). |
| **DUMMY** | Insert dummy cycles | Cycle SCLK without data (count = DUMMY\_CYCLES + EXTRA\_DUMMY). |
| **DATA** | Transfer data | **TX**: Shift FIFO data to flash (writes). **RX**: Capture flash data to FIFO (reads). |
| **STOP\_CS** | End transaction | Deassert CS#; return to IDLE. |

#### ****3. Macronix MX25L6436F Command Support****

| **Command** | **Opcode** | **CMD LANES** | **ADDR LANES** | **DATA LANES** | **Dummy Cycles** | **Direction** |
| --- | --- | --- | --- | --- | --- | --- |
| **4READ (Quad I/O)** | 0xEB | Single (0x0) | Quad (0x2) | Quad (0x2) | 6 (80 MHz) 10 (133 MHz) | Read |
| **QREAD (Quad Output)** | 0x6B | Single (0x0) | Single (0x0) | Quad (0x2) | 8 | Read |
| **PP (Page Program)** | 0x02 | Single (0x0) | Single (0x0) | Single (0x0) | 0 | Write |
| **SE (Sector Erase)** | 0x20 | Single (0x0) | Single (0x0) | None (0x0) | 0 | Write |
| **WREN (Write Enable)** | 0x06 | Single (0x0) | None (0x0) | None (0x0) | 0 | Write |

**Notes**:

* **Dummy cycles** for 4READ depend on the flash’s DC bit (Configuration Register).
* **Quad commands** require QE=1 (Status Register Bit 6).