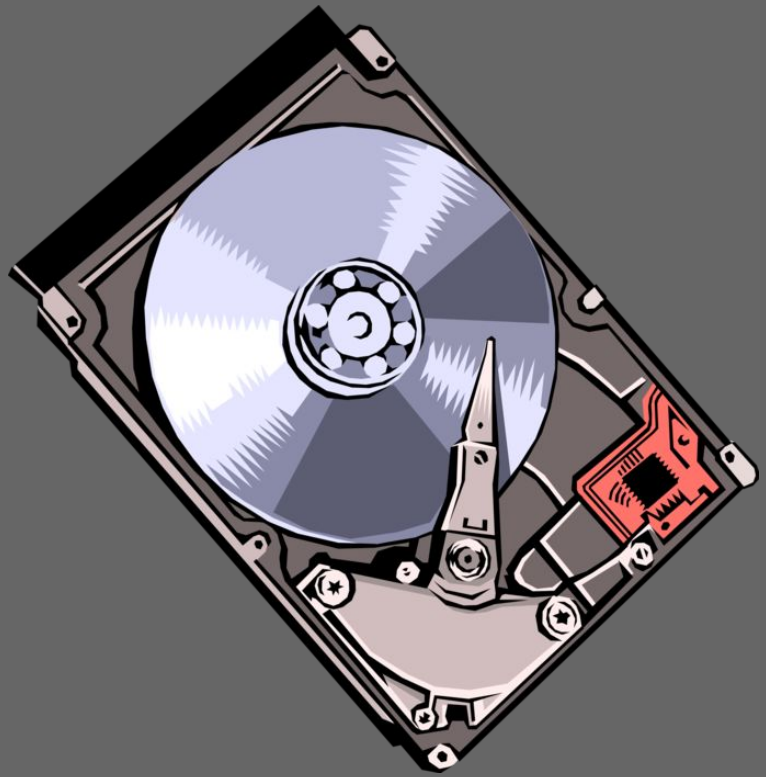
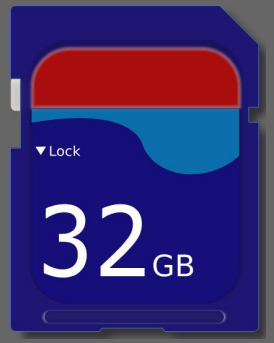
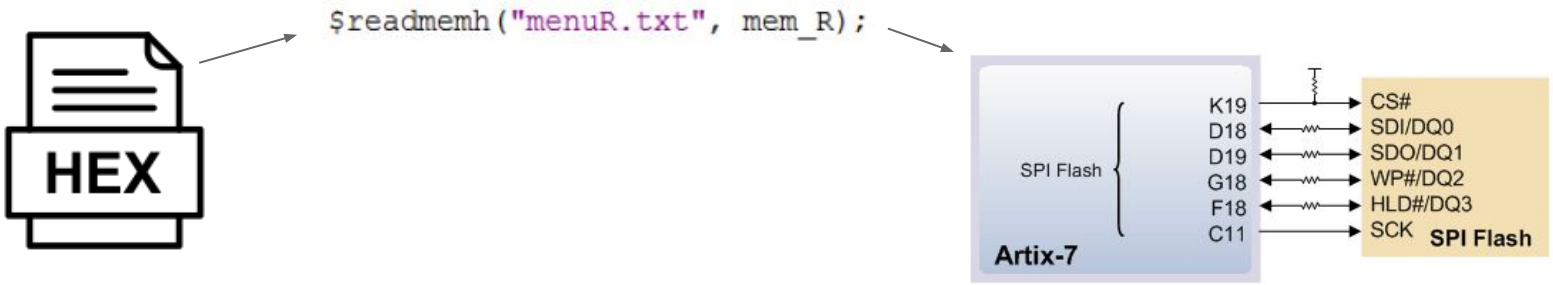
CPE 470 - Non Volatile Storage





The Vivado Way



- FPGAs → Minimally concerned with how data gets on device
  - FPGA vendor already handles flashing bitstream
- ASICs → No pre-built way of programming a chip
  - Have to build your own way of getting data onto chip
- Design Decisions:
  - What kind of external memory is needed?
  - How will it interface with that external memory?

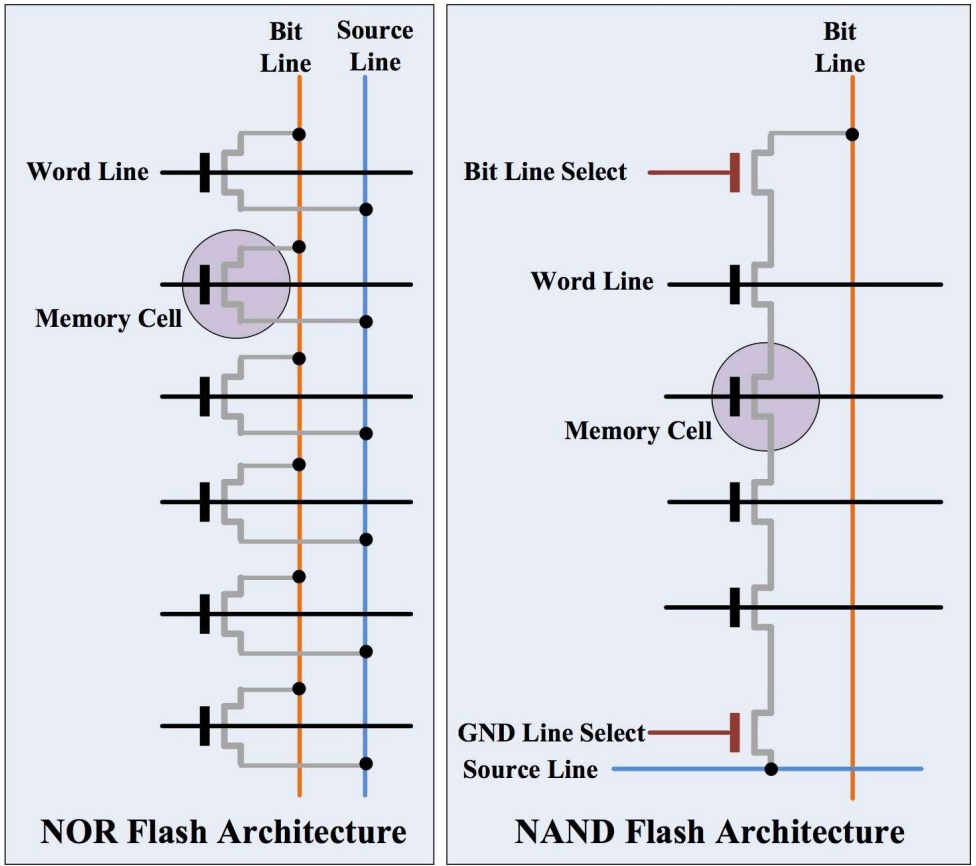
Flash Memory

- NOR Flash
  - Lower Density
  - High Random Access Speed
  - Can write specific words
  - Used in BIOS, program memory
- Nand Flash
  - High Density
  - Lower Random Access Speed
  - Written and read in sequential blocks (like a shift register)
  - Used in SD Cards

Glossary

Flash: transistor-based reprogrammable **NVM**

NVM: Non-Volatile memory, maintains data between power cycles



Flash Tradeoffs

Feature	NOR Flash		NAND Flash	
	General	S70GL02GT	General	S34ML04G2
Capacity	8MB – 256MB	256MB	256MB – 2GB	256MB
Cost per bit	Higher	6.57×10^{-9} USD/bit for 1ku	Lower	2.533×10^{-9} USD/bit for 1ku
Random Read speed	Faster	120ns	Slower	30µS
Write speed	Slower		Faster	
Erase speed	Slower	520ms	Faster	3.5ms
Power on current	Higher	160mA (max)	Lower	50mA (max)
Standby current	Lower	200µA (max)	Higher	1mA (max)
Bit-flipping	Less common		More common	
Bad blocks while shipping	0%		Up to 2%	
Bad block development	Less frequent		More frequent	
Bad block handling	Not mandatory		Mandatory	
Data Retention	Very high	20 years for 1K program-erase cycles	Lower	10 years (typ)
Program-erase cycles	Lower	100,000	Higher	100,000
Preferred Application	Code storage & execution		Data storage	

EEPROM and FRAM

Glossary

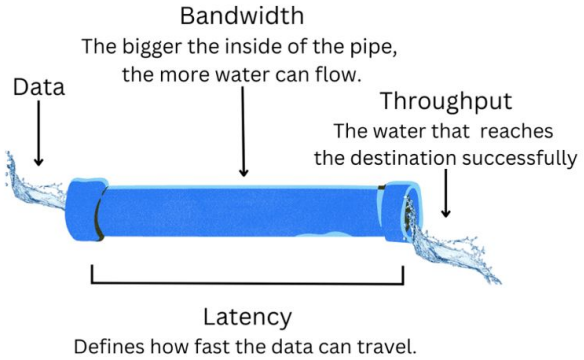
FeRAM: Ferroelectric RAM

EEPROM: electrically erasable programmable read-only memory

- EEPROM
  - Oldest technology
  - Technically flash is a kind of EEPROM
  - Now refers to byte-wise erasable memories
- FeRAM
  - Fastest write speed
  - Lowest density
  - Expensive
  - Lasts a long time

- Both alternatives have niche advantages over flash, but are less mainstream
  - FeRAM for frequently writing smaller data
  - EEPROM for storing single bytes at a time

Throughput



- Factors
  - Frequency
    - What is the clock speed of the bus?
  - Width
    - How many data bits are transmitted per cycle?
  - Efficiency
    - What ratio of the clock cycles is used for actually retrieving data?
    - (Data Cycles) / (Total Cycles) of a transaction
- Calculation
  - Throughput = Frequency * Width * Efficiency Factor
- Metrics
  - Bits per second
    - Gbps, Mbps, etc
  - Bytes per second
    - GBps, MBps, etc
    - Bps is bps / 8

Bus Terminology

Master/Slave Terminology is outdated, to be avoided where possible!
Alternatives Include:

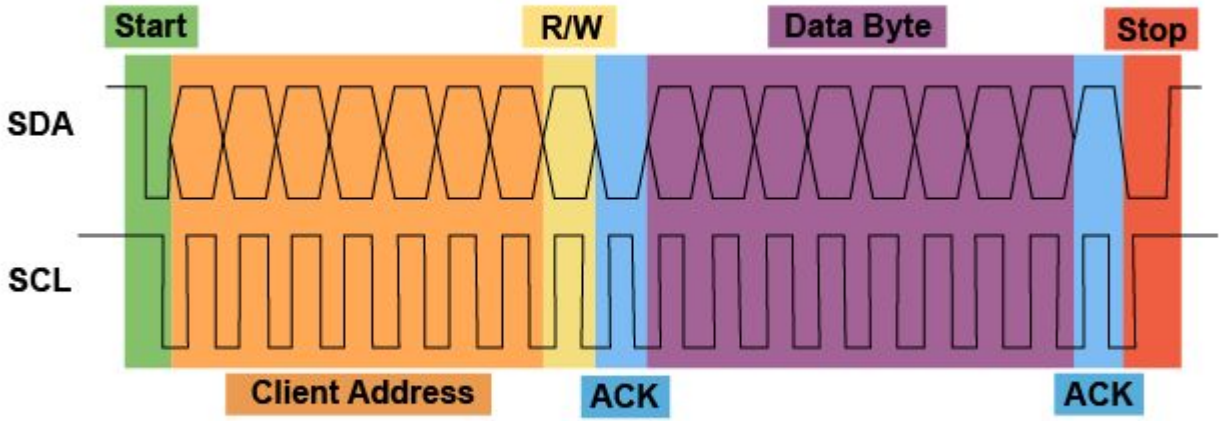
- **Manager / Subordinate**
  - Good because it replaces M/S abbreviations for signals like MOSI/MISO
- **Initiator / Target**
  - Makes sense in systems where a device could be both an initiator or a target
- **Controller / Peripheral**
  - Useful Vocab for when interfacing with Memory Mapped IO
  - Controller accesses peripherals

I2C

Glossary

I2C: Inter-Integrated Circuit

- **2** Wires
  - Bidirectional Data Line, Clock
- **Frequency** = 3.4 MHz
  - Limited by defined protocol speed
- **Width** = 1
- **Efficiency** = Data Width / (Start + Address + Read + Data + Acks + Stop)
  - Shown Example: 8 bit address, 8 bit word
  - Efficiency = 8 / 21 = 38%
  - **Throughput** = 3.5 * 1 * 38% = 1.3 Mbps

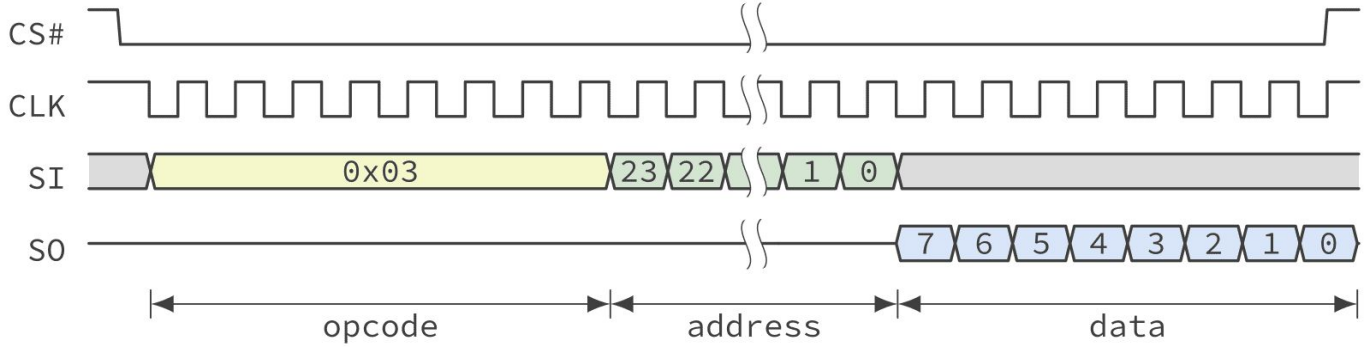


SPI

Glossary

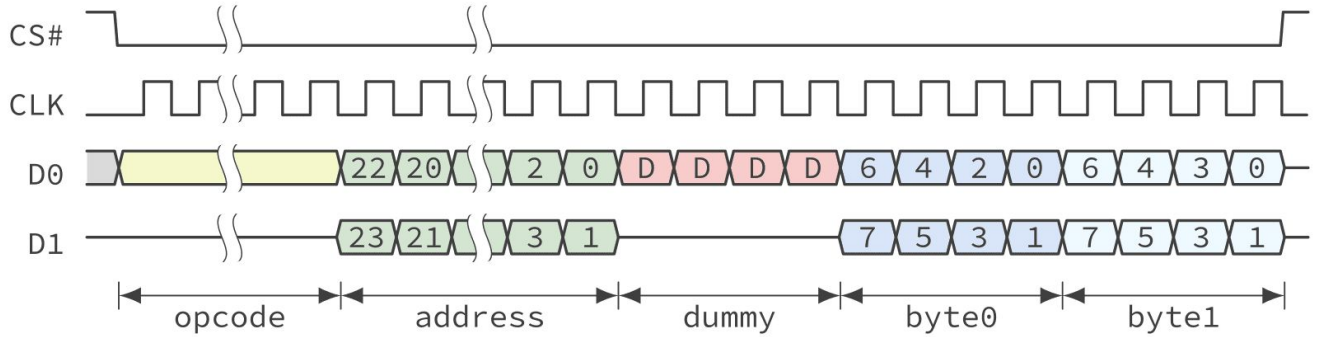
SPI: Serial Peripheral Interface

- **4** Wires
- Purely unidirectional
  - Separate input and output lines, but still 1 bit wide
- **Frequency** = ~40 MHz
- **Width** = 1
- **Efficiency** = Data Width / (Data + Address + Opcode Widths)
  - Example: 8 cycle opcode, 24 cycle address, 32 cycle data word
  - Efficiency = 32 / (8 + 24 + 32) = 0.5
  - **Throughput** = 20 Mbps



Dual SPI

- Still **4** Wires
- Differences:
  - Now Bidirectional
    - Both wires used by both controller and peripheral
  - Opcode is not split among wires
  - Need dummy cycles between address and data so flash can obtain the data
- **Frequency** = ~40 MHz
- **Width** = 2
- **Efficiency** = Data Width / (Data + Address + Opcode Widths)
  - Example: 8 cycle opcode, 12 cycle address, 4 dummy cycles, 16 cycle bit data word
  - Efficiency = 16 / (8 + 12 + 16) = 0.4
    - Diminishing Returns on efficiency due to opcode and dummy data
  - **Throughput** = 32 Mbps



QSPI, OSPI, and Beyond

- What if we keep adding parallel wires?
  - 2 + N wires (Quad SPI ⟶6 wires, Octal SPI ⟶10 Wires)
- Twice the wires gives almost twice the performance
  - Start seeing diminishing returns
  - Opcode and dummy time begins to dominate

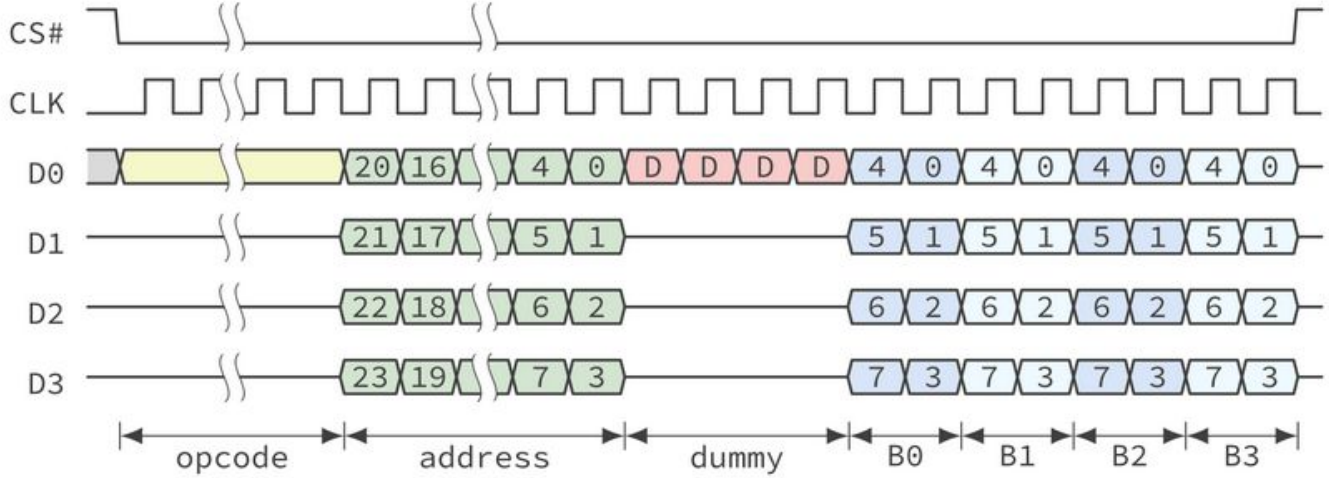


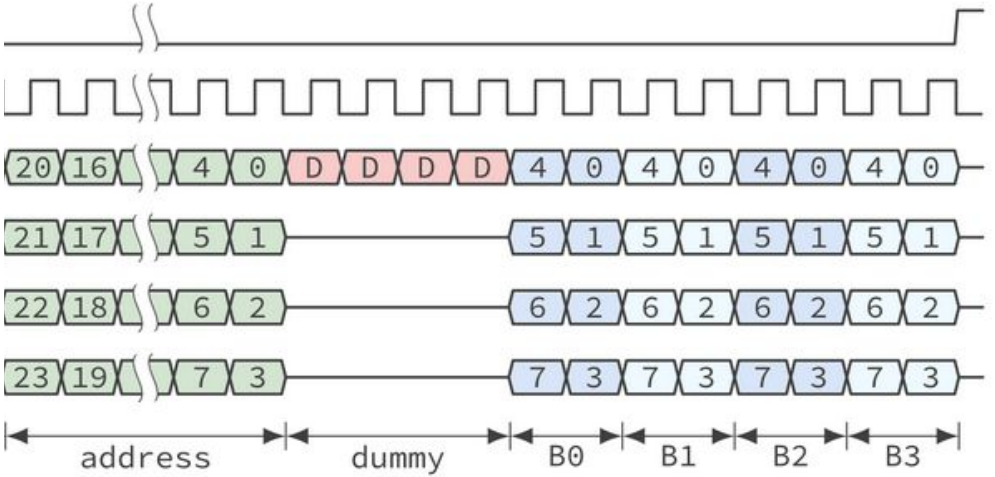
Figure 9 – Quad I/O fast read.

SPI Variations: XIP, DDR

Glossary
XIP: eXecute In Place
DDR: Double Data Rate

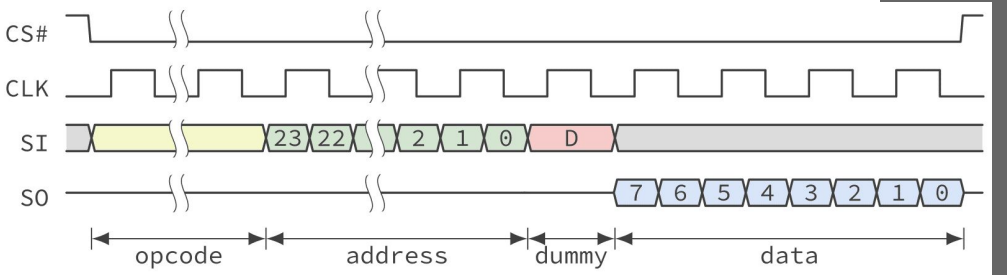
XIP Variation

- As we parallelize, serial opcode begins to dominate
  - Solution → Get Rid of Opcode
- Place Flash chip in XIP mode: read only
  - Implied Read → No Need for Opcode
- Often used for processor instruction memory
  - Usually never going to write

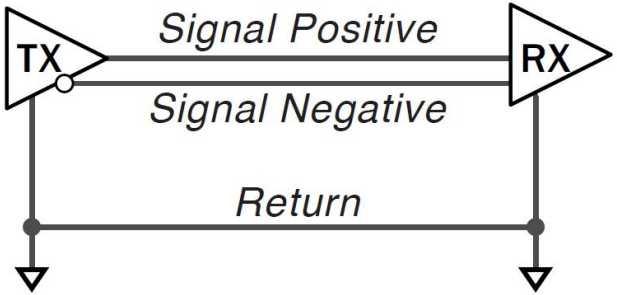


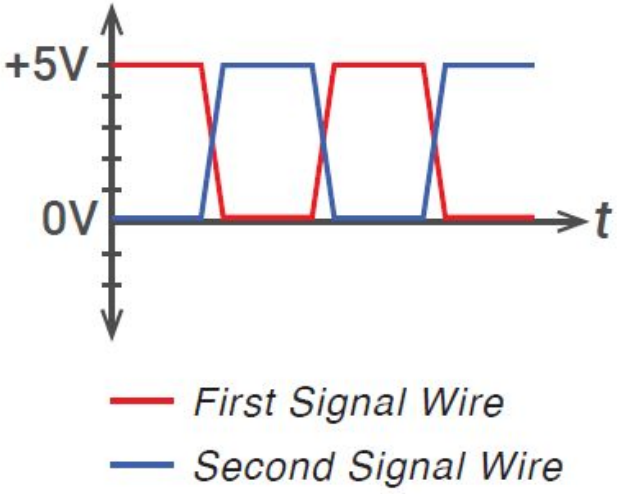
DDR Variation

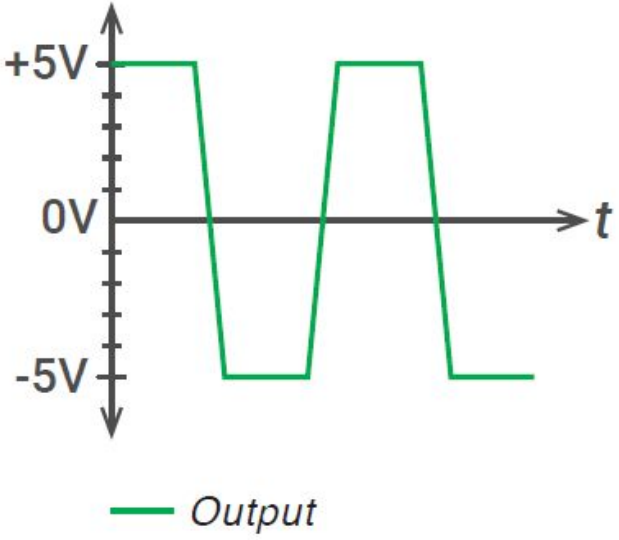
- Change data on both clock edges
- Effectively doubles throughput
  - except for dummy time, which is a fixed delay



Differential Signals







SATA

SATA Pinout - Plug

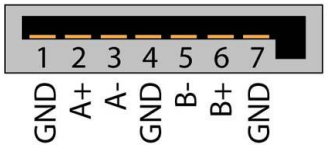
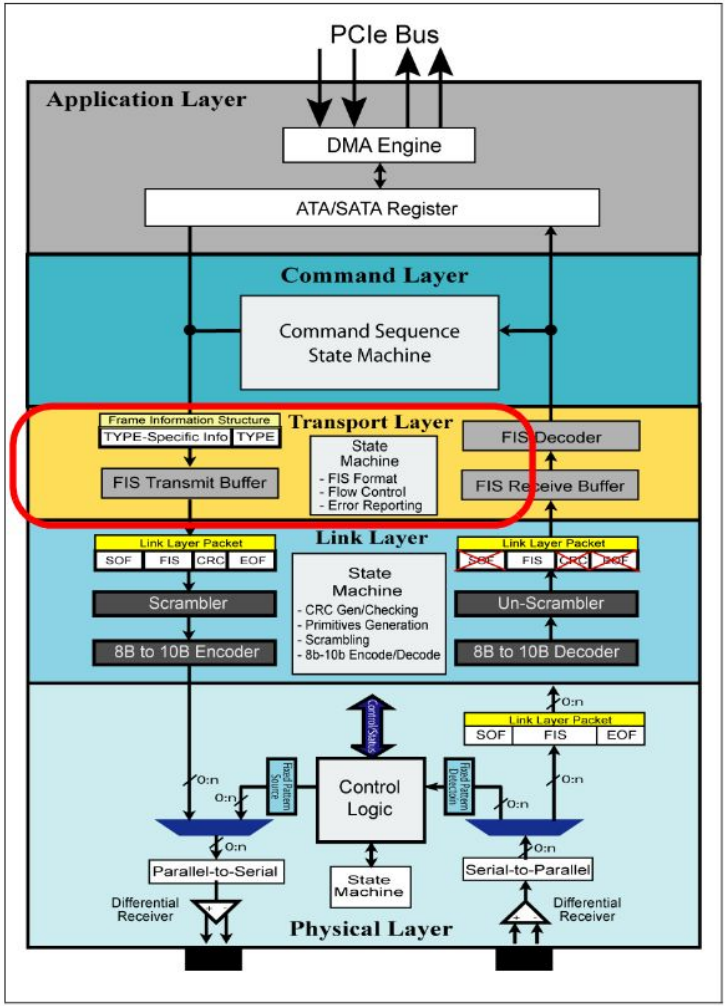


Figure 4-2: Transport Layer



References

- https://www.jblopen.com/qspi-nor-flash-part-3-the-quad-spi-protocol/
- https://www.embedded.com/flash-101-nand-flash-vs-nor-flash/
- https://www.design-reuse.com/articles/41861/execute-in-place-xip-nor-flash-spi-protocol.html
- https://www.digikey.com/en/articles/the-fundamentals-of-embedded-memory
- https://sparxeng.com/blog/hardware/mastering-differential-signals
- https://www.mindshare.com/files/ebooks/SATA%20Storage%20Technology.pdf