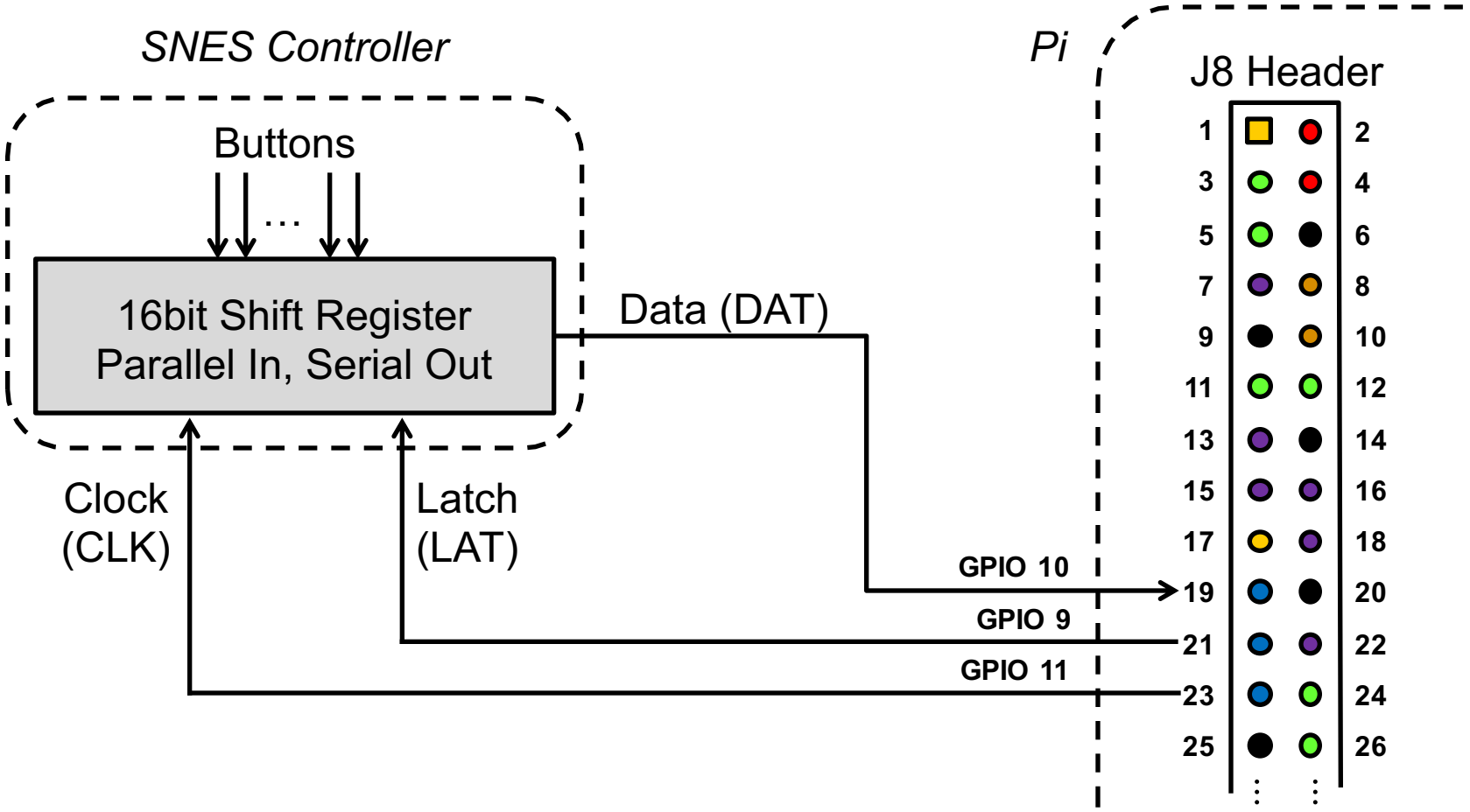
CPSC 359 – Tutorial #6

SNES Controller & Timer Interface

Originally from Andrew Kuipers

Updated for RPi3 / Winter 2018

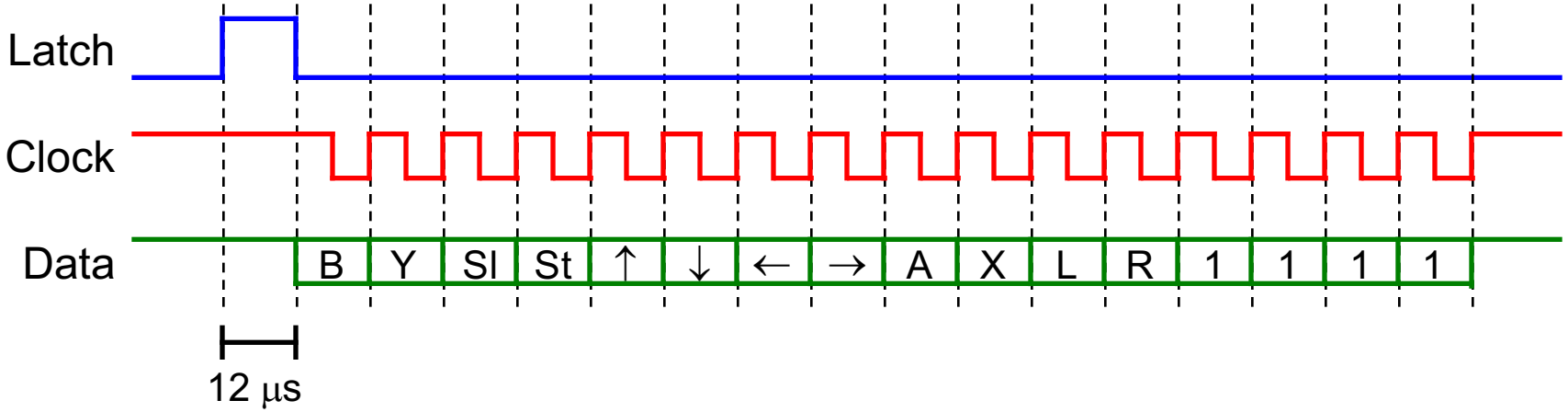
SNES Controller



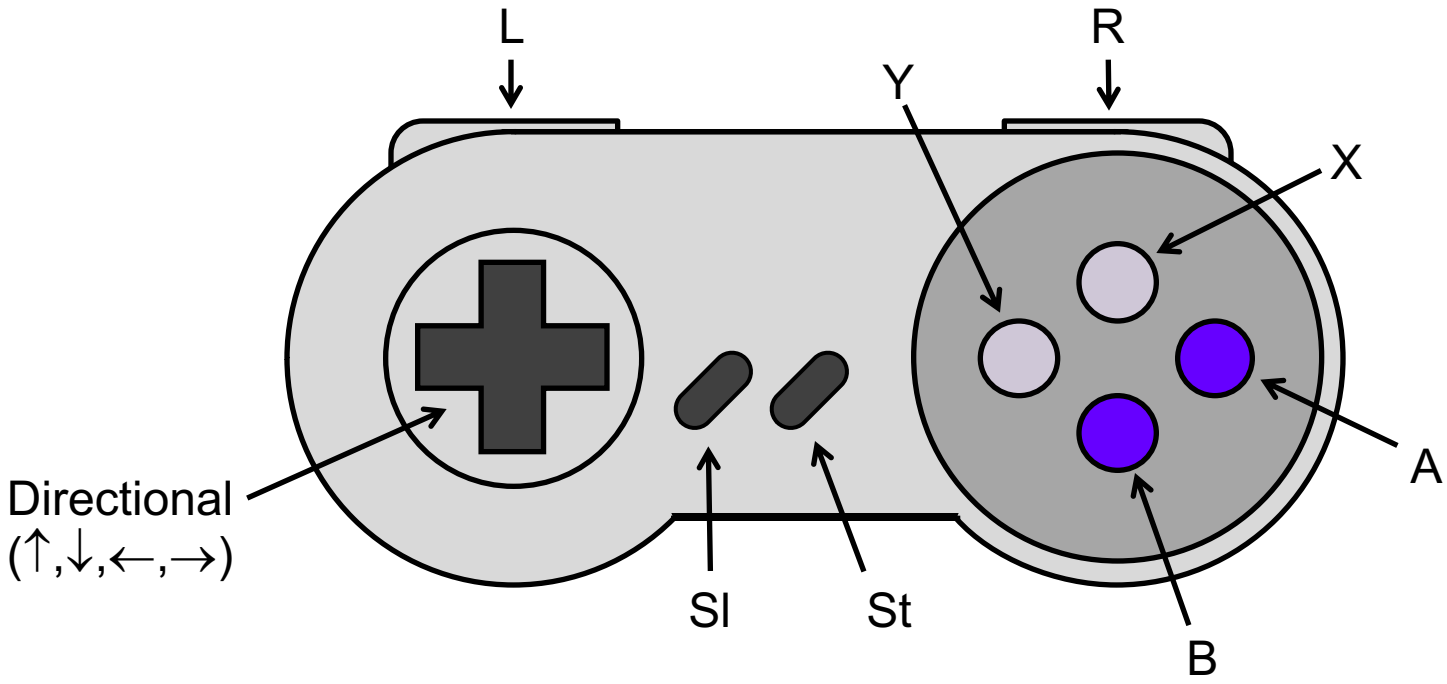
SNES Controller

SNES Controller Protocol

- Set *Latch* line high for 12 μs to latch button states in shift register
- Cycle *Clock* line 16 times to shift data out; 6 μs high, 6 μs low
- Read *Data* line each time Clock set low
  - 0 = button pressed, 1 = button not pressed



SNES Controller



Microseconds Delay

- For delaying in microseconds, you might use the following function in your code:

```
mov     r0, #12          @ for 12 μs delay
bl      delayMicroseconds
```

- In bare metal programing, you need to design your own timer.

Timer Interface

Simple Timer

- Wait in a loop until *current time* = *original time* + *delay*
- Can't execute other instructions until loop exits

Advanced Timer

- Load a Timer Compare register with *original time* + *delay*
- Create an Interrupt Service Routine (ISR) for the Timer Interrupt
- Continue execution of other instructions until the interrupt fires
  - Need to consider concurrency!

Timer Interface

Timer Interface

- System Timer interface at address 0x3F003000 (BCM2835, p. 172)
- Increments a 64bit counter at one micro-second (μs) intervals

Offset	Reg. Name	Description
0x00	CS	Timer Control / Status
0x04	**CLO**	**Timer Counter (Low 32 bits)**
0x08	**CHI**	**Timer Counter (High 32 bits)**
0x0C	C0	Timer Compare 0
0x10	C1	Timer Compare 1
0x14	C2	Timer Compare 2
0x18	C3	Timer Compare 3

Timer Interface

CLO – Timer Counter (Low 32 bits) (BCM2835, pg. 173)

- Stores the low 32bits of 64bit current time counter
- Incremented at 1 μs intervals by the system timer clock
- Read and compare against this value to implement Simple Timer
- 2^{32} μs ≈ 1.2 hours

CHI – Timer Counter (High 32 bits) (BCM2835, pg. 173)

- Stores the high 32bits of 64bit current timer counter
- Incremented whenever CLO overflows 32bits (and resets to 0)

Timer Interface

CS : Timer Control / Status (BCM2835, pg. 173)

- Low 4 bits are match status bits, high 28 bits unused
- When a bit is set by the system timer, an interrupt is generated
- Match status bits are cleared by writing a 1 (setting) that bit

C0 – *C3* : Timer Compare *n* (BCM2835, pg. 174)

- When *CLO* = *Cn*, set match status bit *n* in *CS*
- Interrupt Service Routine will need to clear the appropriate status bit
- *Note: C0 and C2 used by GPU, only use C1 and C3*