CSSE2010

CSSE2010/CSSE7201
Lecture 19

More I/O

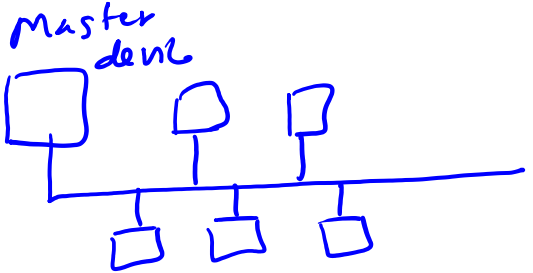
School of Information Technology and Electrical Engineering
The University of Queensland

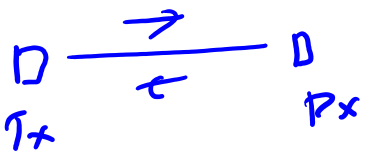
Today

- More I/O
  - I^2C, TWI, SPI
    - Synchronous Serial
- AVR I/O coding
  - Volatile variables
  - Setting/clearing bits in I/O registers

Bus Masters and Slaves

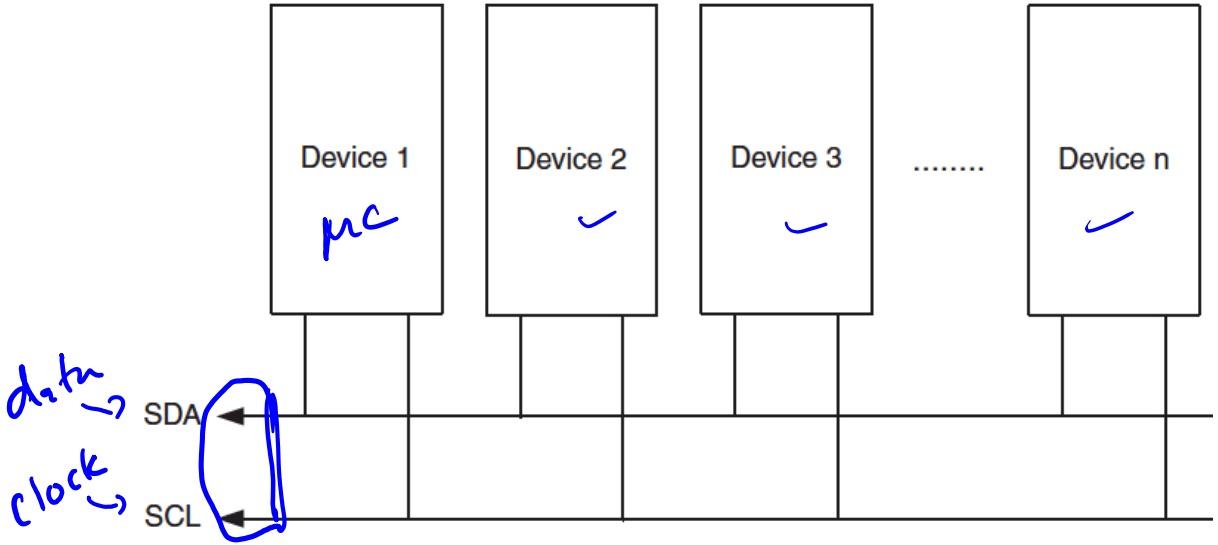
- Master
  - Device which initiates (and controls) the transfer of data
- Slave
  - Waits for requests
  - Responds to master
- Some devices can be both masters and slaves (at different times)
  - Memory can never be a bus master





I²C, TWI

- I²C = Inter-Integrated Circuit
  - created by Philips Semiconductor (now NXP)
- TWI = Two-wire Interface
  - Name used by Atmel

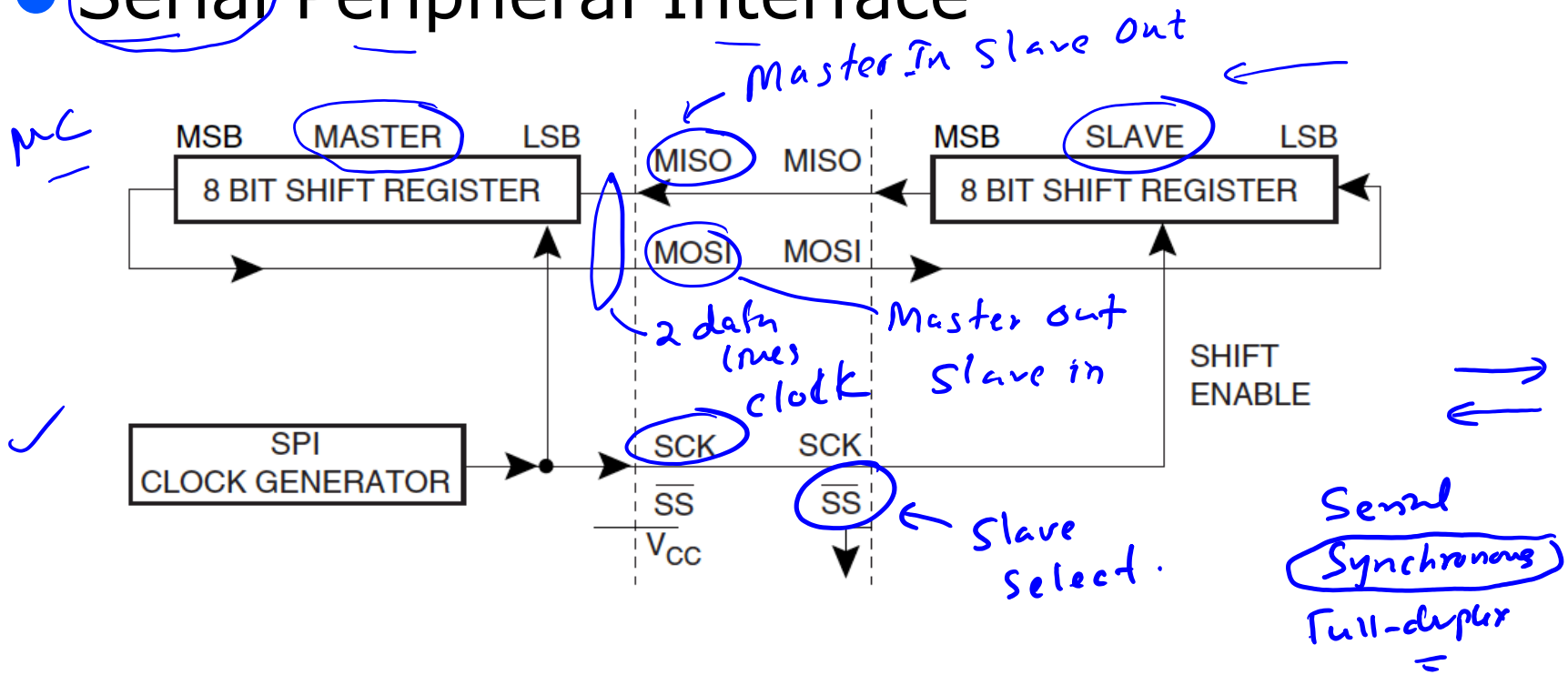


TWI on AVR

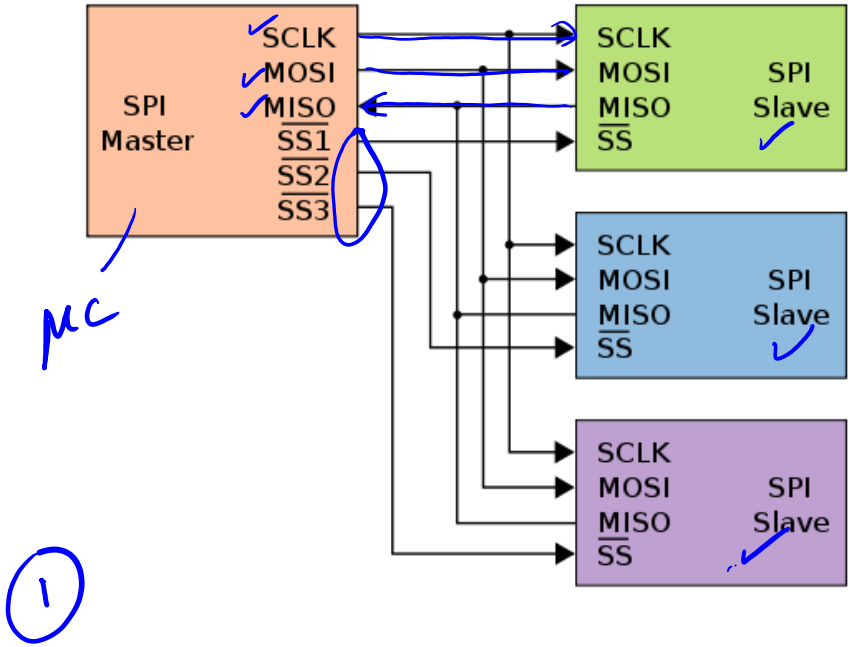
- ATmega324A has one TWI port
  - Alternate use for pins
    - C0 (SCL)
    - C1 (SDA)
- See datasheet pages 211 to 239 for details
  - Includes full description of TWI

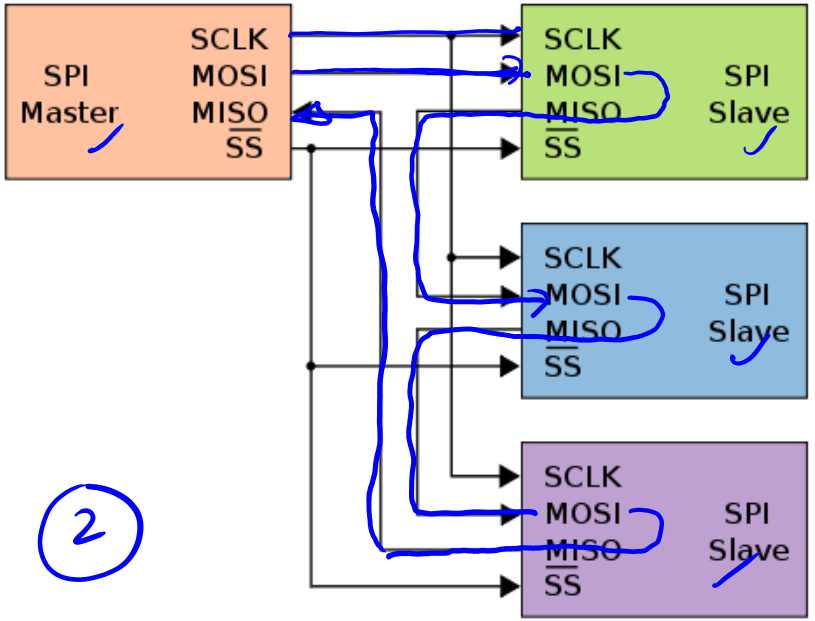
SPI

- Serial Peripheral Interface



SPI – Other configurations

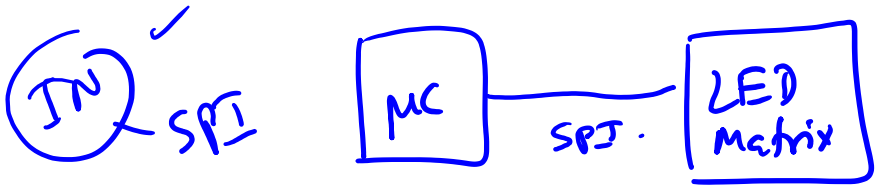




Images by Colin M.L. Burnett

SPI on the AVR

- ATmega324A has one SPI port
  - Shares pins with upper half of port B
  - Hardware takes care of generating signals
  - Can be master or slave
  - See datasheet pages 166 to 174
  - USARTs can also be used as SPI masters

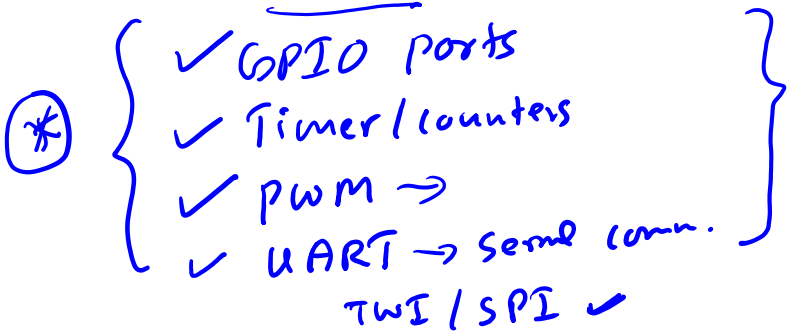


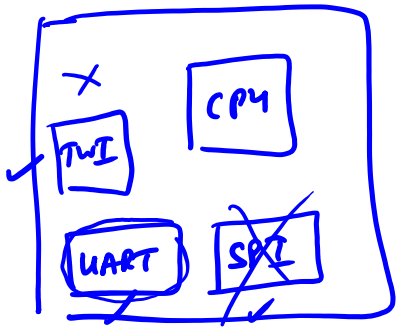
SPI on the AVR (cont.)

- For Master operation, can choose clock rate divider:
  - 2 (4MHz @ 8MHz clock)
  - 4 (2MHz)
  - 8 (1MHz)
  - 16 (500kHz)
  - 32 (250kHz)
  - 64 (125kHz)
  - 128 (62.5kHz)
- For slave operation, fastest clock is f_{OSC} / 4 (2MHz in our case)

Bit Bashing

- Sometimes microcontroller doesn't have support for a bus standard
  - Or enough interfaces of a particular standard
- Can simulate I/O standards using general purpose ports





AVR I/O Coding
Volatile variables

- Example: `volatile int running;`
- Tells compiler that variable can change outside of standard flow of control
  - e.g. in interrupt handler or a thread
- Prevents compiler applying optimisations
- Example:

```
running = 0;
while(!running) {
    ; /* Do nothing */
}
```

Without volatile, optimising compiler will change this to

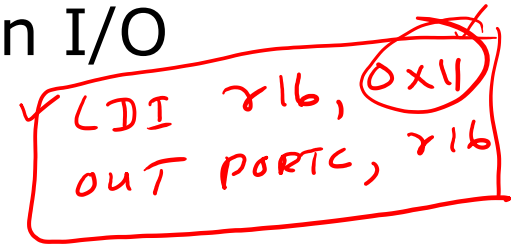
```
while(1) {
    ;
}
```

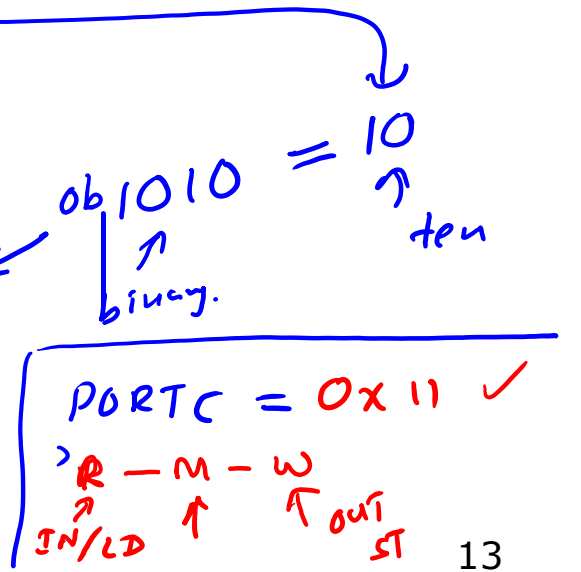
AVR I/O Coding
Setting I/O register bits

- When setting/clearing specific bits in I/O registers, be explicit about the bits

Example (all have the same effect):

- Poor:
  - `TCCR1B = 10;`
- Better:
  - `TCCR1B = 0x0A;`
- Best:
  - `TCCR1B = (1<<WGM12)|(1<<CS11);`
- Makes code easier to understand





Setting I/O register bits

- Where do names come from?
  - See datasheet (note errors – see IO register summary)
- e.g. for TCCR0B (page 112)

Bit	7	6	5	4	3	2	1	0
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

- Bit names are defined as macros, e.g.

```
#define WGM02 3
```

- Can use same names in assembly language also, e.g.

```
ldi r16, (1<<WGM02)|(1<<CS01)
```

assembler converts this to a constant.

Clearing/Toggling I/O register bits

- Same rules apply
- To clear a bit, e.g.

```
PINC &= ~(1<<5);
```

- To toggle a bit, e.g.

```
/* Enable Output Compare A Interrupt
** for timer 2 */
TIMSK2 ^= (1<<OCIE2A);
```