CSSE2010

CSSE2010/CSSE7201
Lecture 14

More C
C Programming on the AVR

School of Information Technology and Electrical Engineering
The University of Queensland

Admin and Reminders

- Assignment 1 – part 1 marks will be released this week Friday
- Assignment 1 – part 2 marks will be released next week.
- Quiz 6 is due this week Friday

Remaining Labs of this Course

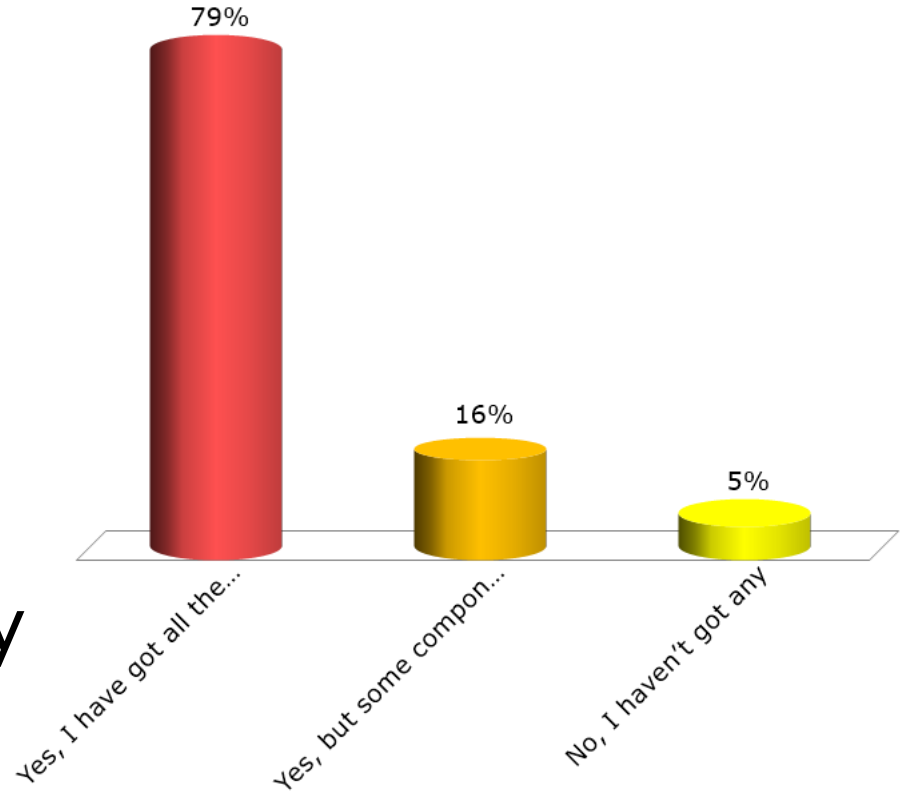
	Week 1-7 Digital Logic	Week 8-13 AVR Microcontroller
IN Students	Logic chips and breadboard circuits	ATmega324A microcontroller and other peripherals in the lab kit. Programming through Microchip Studio and Pololu USB programmer
EX Students	Logisim simulations	Arduino Uno (ATmega328P), breadboard and other components and peripherals you should have acquired by now. Programming through Microchip Studio and Arduino in-built programmer

EX Students: Have you acquired your Arduino based lab hardware components?

A. Yes, I have got all the components

B. Yes, but some components are missing, but I have the Arduino Uno

C. No, I haven't got any



AVR Assembly Instructions (last time)

ADD
ADC
SUB
SBC
AND
OR
FOR
DEC
INC
SER
CLR

LSL
LSR
ASR
ROR
ROL

BREQ
BRNE
:
JMP
RJMP

Control flow

IN
OUT
LD
LDS
ST
STS

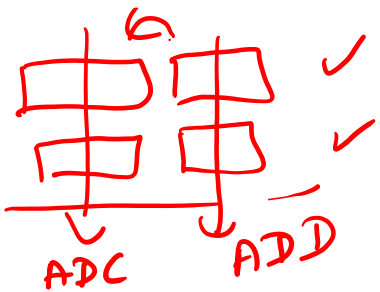
Mem Access

CALL
RCALL
RET
RETI

calls & return

Data types in ISA (last lecture)

- Other data types can be supported through software!
  - 16-bit integer operations can be built out of 8-bit operations, using a sequence of instructions.
  - e.g. AVR add 16-bit quantity r5:r4 to r11:r10



Bit shifts in C

- Recall `lsl`, `lsr`, `asr` from last lecture

C equivalents are:

- `a << b` means a shifted left by b bits
- `a >> b` means a shifted right by b bits
  - Performs logical shift right if a is unsigned
  - Performs arithmetic shift right if a is signed

unsigned → lsr

signed → asr

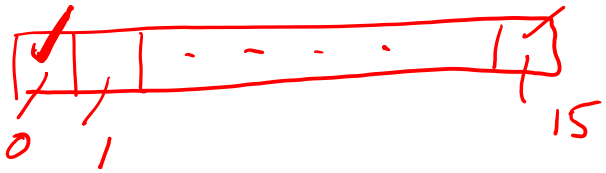
Arrays in C
(Were mentioned in Lab 9)

- Declaring an array

```
type variable_name[size];
```

  - Examples:

```
char message[16];
int values[10];
```



- Accessing elements within an array

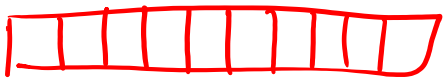
```
variable_name[index]
```

  - index = 0 … size-1
    - This is called zero-based indexing
  - Examples:

```
message[0] = 'c';
values[9] = values[8]++;
```

character constant

post incr.

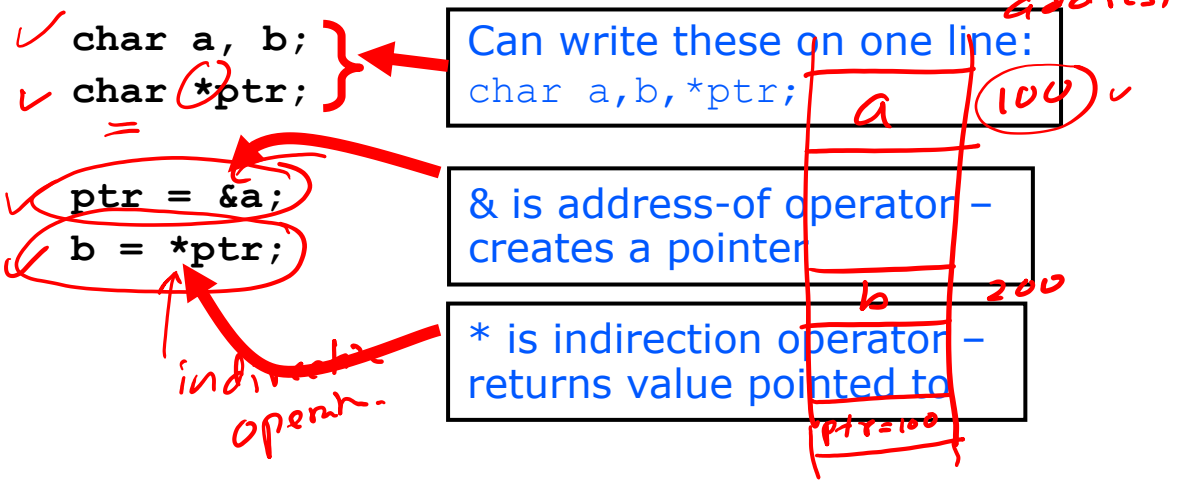


a++

++a

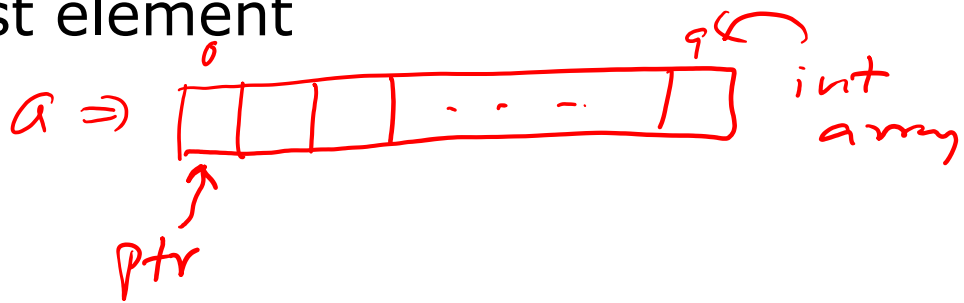
Pointers

- C has concept of pointers
- Pointer declaration
  - ***type * variable_name;***
  - *variable-name* is a pointer to something of given *type*
    - How? – pointer variables store memory addresses
- Example:



Pointers and Arrays

- Array name can be treated as a pointer to the first element
  - i.e. address of first element
- Example:

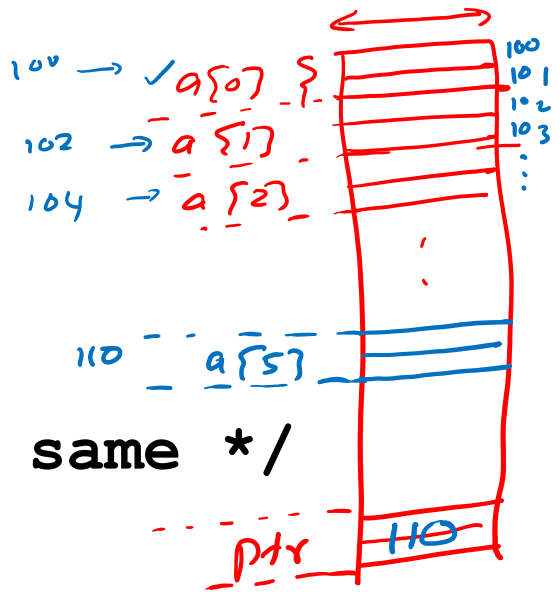



```
int a[10];
int *ptr;

  /* following statements are same */
ptr = a;
ptr = &a[0];
```

Operations on Pointers

- Addition/subtraction operations on pointers work in multiples of the size of the object being pointed to
- Example

```
int a[10];
int *ptr;

/* following statements are same */
ptr = a+5;
ptr = &a[5];
```



Traversing an Array

- Two examples of clearing an array
- Using **index**:

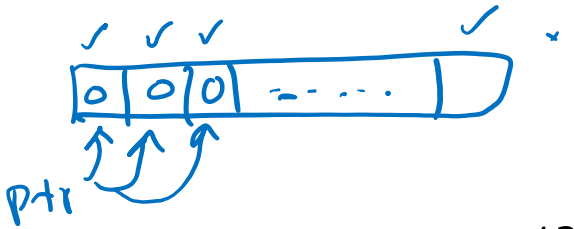
```
int a[10];
int index;
for(index=0; index<10; index++) {
        a[index] = 0;
}
```

index = index + 1

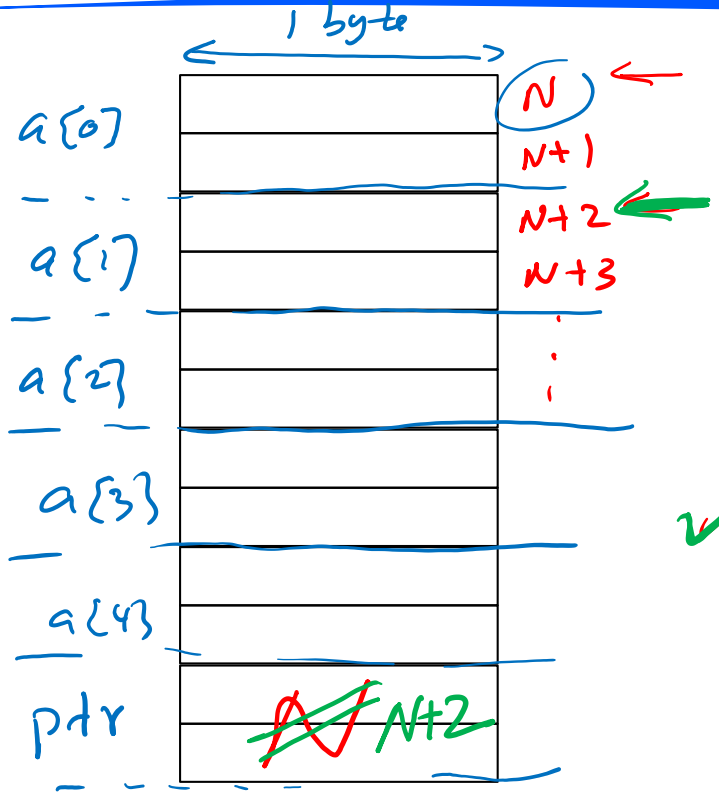
- Using **pointer**:

```
int a[10], *ptr;
for(ptr=a; ptr < a+10; ptr++) {
        *ptr = 0;
}
```

Adding one to an array pointer makes it point to next element in array



Arrays in Memory



int a[5]

our integers are 16 bits

int *ptr

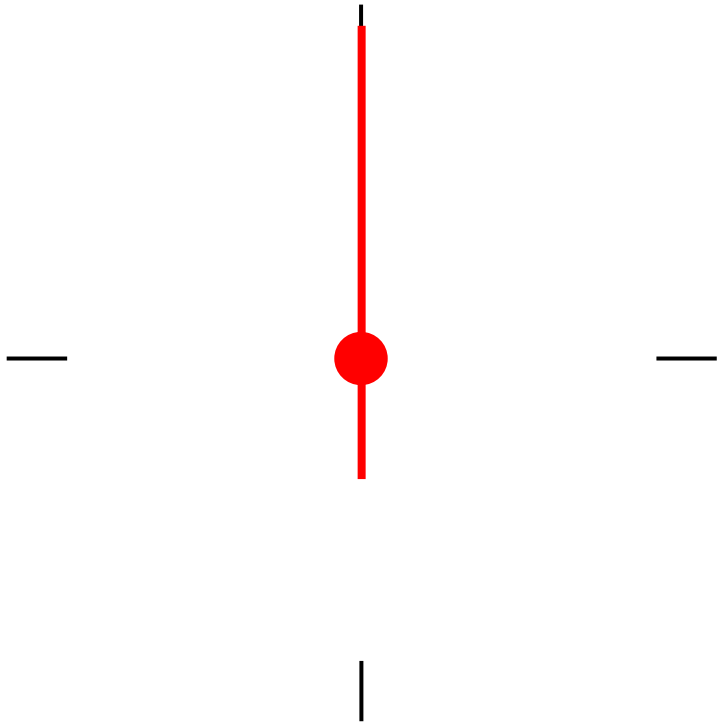
ptr = a ✓

ptr++ ⟹ ptr = ptr + 1

Figure to be completed in class

Short Break

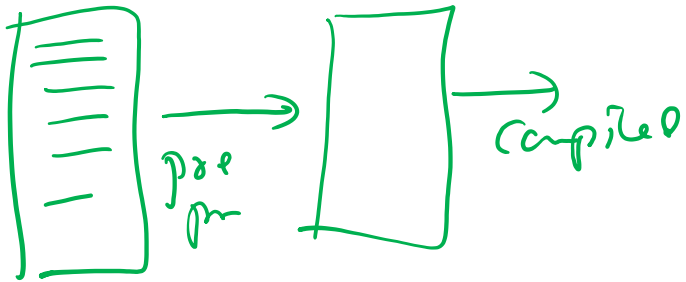
- Stand up and stretch



Preprocessor

- C programs are first parsed by a preprocessor
- Preprocessor replaces preprocessor directives with substitute text
- Examples
  - `#include` …
  - `#define` … (macros)
  - many others

```
#if
#ifdef
#else
#undef
…
```



include <stdio.h>

Header (.h) files

- Header files (.h files) are used for common macros, function prototypes etc
  - Like m324Adef.inc for assembly
- Examples
  - stdio.h – includes prototypes of printf() etc
    - AVR has a simplified version of this (e.g. no files)
      - printf() will only work if you specify where the characters go, e.g. sent via serial port
  - avr/io.h – AVR has this to define useful constants and functions/macros
    - Definitions depend on the actual device targeted

Preprocessor Examples

System header files

```
#include<filename.h>
#include"filename.h"
```

Macros – textual substitutions

```
#define pi 3.1415
#define EVER ; ;
#define sum(a,b) (a+b)
```

$y = x+3 + a+2$

$z = sum(x,3) \Rightarrow z = x+3$

$y = sum(x,3) + sum(a,2)$

Pi

```
for( ; ; )
for(EVER)
```

Given

```
#define SQUARE(x) (x*x)
```

What is the result of `SQUARE(1+2)`?

x ≡ 1+2

1+2*1+2

1+2+2 = 5

0% 1. 1

0% 2. 2

0% 3. 3

3% 4. 4

36% 5. 5

3% 6. 6

0% 7. 7

0% 8. 8

58% 9. 9

#define square(x) ((x)*(x))

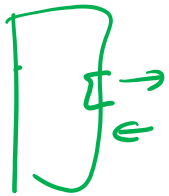
(1+2)*(1+2)

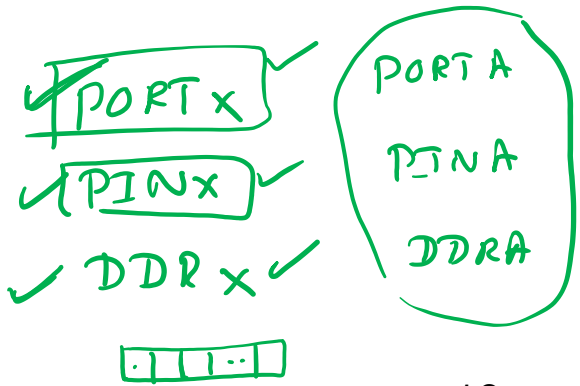
3*3 = 9

0

AVR C Macro Examples

```
#define _MMIO_BYTE(mem_addr) \
                (*(volatile uint8_t *)(mem_addr))
#define _SFR_IO8(io_addr) \
                _MMIO_BYTE((io_addr) + 0x20)

/* Input Pins, Port A */
#define PINA    _SFR_IO8(0x00)

/* Data Direction Register, Port A */
#define DDRA    _SFR_IO8(0x01)

/* Data Register, Port A */
#define PORTA   _SFR_IO8(0x02)
```





AVR C Macro Examples (cont.)

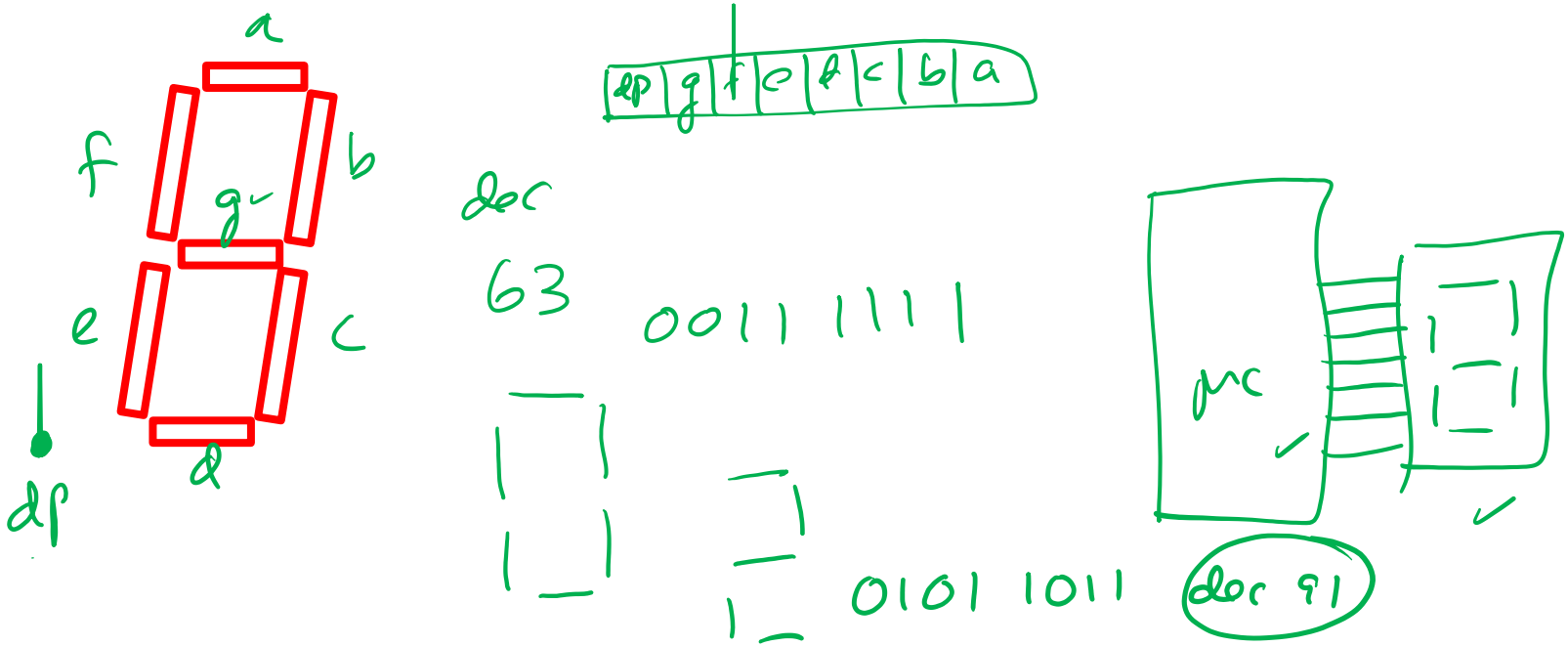
- Upshot of these macro definitions…
  - I/O register names can be used as variables, e.g.

```
DDRA = 0xFF;
```

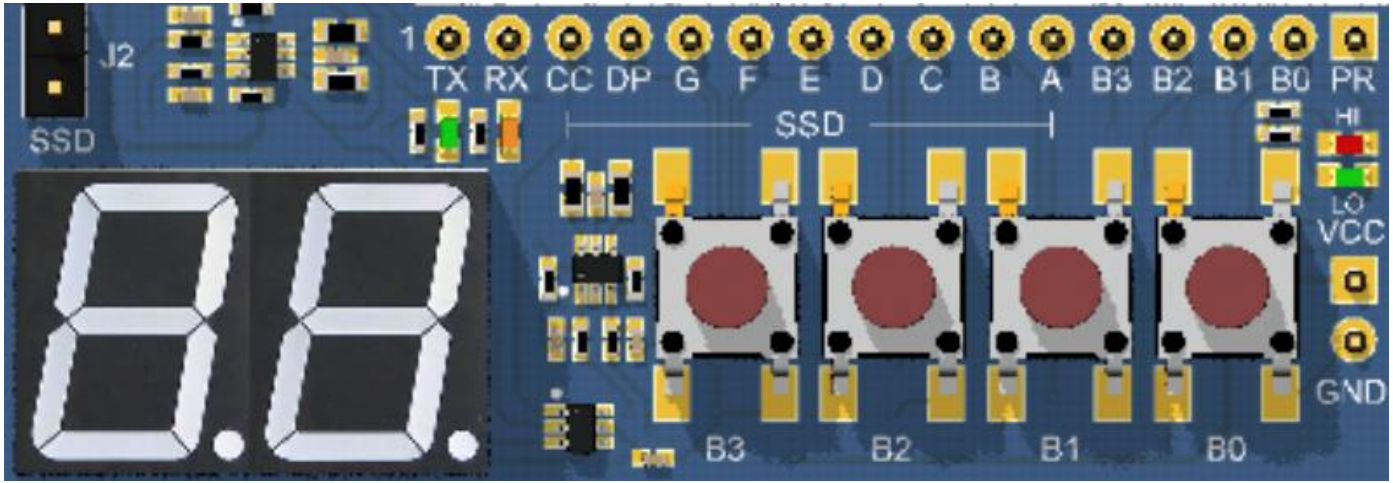
```
PORTA = PINC | 0xF0;
```

Seven Segment Display Programming Example

- To be covered in class



Seven Segment Display on IO Board



FD students

EX students: you will interface your 2-digit seven segment display to the Arduino Uno using the breadboard.

```
#include <avr/io.h>

/* Seven segment display values */
uint8_t seven_seg[10] = { 63,6,91,79,102,109,125,7,127,111};

int main(void) {
   uint8_t digit;

   /* Set port A pins to be outputs, port C pins to be inputs */
   DDRA = 0xFF;
   DDRC = 0;     /* This is the default, could omit. */
   while(1) {
      /* Read in a digit from lower half of port C pins */
      /* We read the whole byte and mask out upper bits */
      digit = PINC & 0x0F;
      /* Write out seven segment display value to port A */
      if(digit < 10) {
          PORTA = seven_seg[digit];
      } else {
          PORTA = 0;
      }
   }
}
```