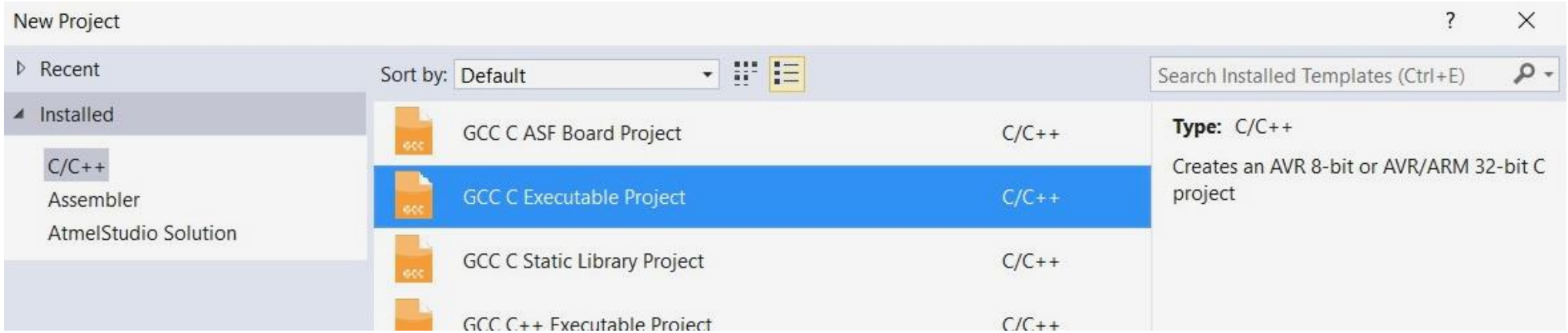
C Project

Ensure you select the ‘GCC C Executable Project’ type

This will matter in next weeks lab as we start to interface with more complex subsystems of the lab board



Setting ADC Registers

- ADCSRA
  - Enables functionality of the ADC
  - Gives us a flag for checking to see when a conversion is complete
- ADCSRB
  - Sets High Speed Mode and Auto Trigger source
  - For now, don't worry about this register (it's default values are fine!)
- ADMUX
  - Selects the voltage reference for the ADC (Use AVCC – Bit7:6 = 11)
  - Selects the Left-Adjust for the ADC data – Enable this – Bit 5 = 1
  - Selects the ADC channel/pin for checking – Bits 4:0 set depending on ADC input

Setting ADC Registers

Analog to Digital Converter Registers

ADC Data Register – ADCL and ADCH

ADLAR = 0: (right adjusted)

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	

ADLAR = 1: (left adjusted)

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	

When an ADC conversion is complete, the result is found in these two registers. The two registers can be accessed by using ADCH and ADCL register names, or if wanting to access them both use the ADC register name. If the result is left adjusted (see the ADLAR bit in the ADMUX register) and no more than 8-bit precision (7-bit + sign bit for differential input channels) is required, it is sufficient to read ADCH. (Note reading just the ADCL will prevent the ADC Data Register from being updated).

ADC Control and Status Register B – ADCSRB

Bit	7	6	5	4	3	2	1	0	
	ADHSM	ACME	-	-	-	ADTS2	ADTS1	ADTS0	ACSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

• Bit 7 – ADHSM: ADC High Speed Mode
Writing this bit to one enables the ADC High Speed mode. This mode enables higher conversion rate at the expense of higher power consumption.

• Bit 2:0 – ADTS2:0: ADC Auto Trigger Source
If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS2:0 settings will have no effect

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free running mode
0	0	1	Analogue Comparator
0	1	0	External interrupt 0
0	1	1	Timer 0 compare match
1	0	0	Timer 0 overflow
1	0	1	Timer 1 compare match
1	1	0	Timer 1 overflow
1	1	1	Timer 1 input capture

ADC Control and Status Register A – ADCSRA

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA $06 ($26)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Bit 7 ADEN: ADC Enable
Setting this bit enables the ADC, clearing the bit turns the ADC off.

Bit 6 ADSC ADC Start Conversion
Set this bit to start each conversion. When the conversion is complete, the bit returns to zero. Note in Free Running Mode, set this bit to start the first conversion.

Bit 5 ADATE ADC Auto Trigger Enable
When this bit is zero, the ADC is in single conversion mode, when it is one, auto triggering is enables.

Bit 4 ADIF ADC Interrupt Flag
This bit is set when an ADC conversion is completed. The bit is cleared by hardware when the interrupt routine is executed. Alternatively, writing a logical one to the flag can clear the bit.

Bit 3 ADIE ADC Interrupt Enable
When this bit is set and the I-bit in SREG is set, the ADC Conversion Complete Interrupt is activated.

Bits 2:0 ADPS2: 0 ADC Prescaler Select Bits
These bits determine the conversion speed of the ADC related to the system clock.

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ADC Multiplexer Selection Register – ADMUX

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX $07 ($27)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

Bit 7:6 (REFS1: 0) Reference Selection Bits
These are used to alter the voltage reference selection of the ADC.

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

Bit 5 ADLAR ADC Left Adjust Result
Writing a one to this bit will left adjust the ADC conversion result in the ADC data register, other wise the result is right adjusted.

Bits 4:0 MUX4: 0 Analog Channel and Gain Selection Bits
This selects which channel will be converted, and whether they are single or differential and which gain setting is used.

	Single Ended	Positive Differential	Negative Differential	
MUX4.0	Input	Input	Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000	N/A	ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010		ADC0	ADC0	200x
01011		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110		ADC2	ADC2	200x
01111		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x
11101		ADC5	ADC2	1x
11110	1.22 V (VBG)	N/A		
11111	0 V (GND)			

Obtaining ADC Values for Single-Conversion

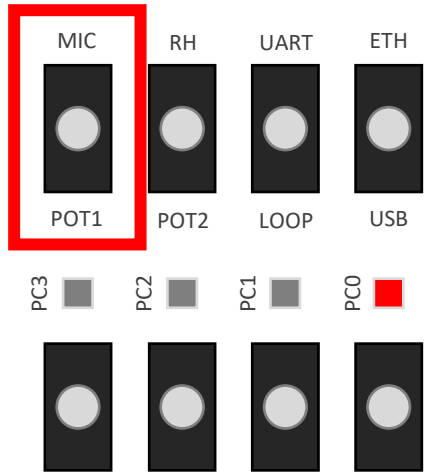
- Use a function to return the value of the ADC
  - The example to the right is also used extensively in your textbook.
- Define the ADC input by passing ADMUX as the channel parameter
  - For each ADC input, define a separate ADMUX value
  - In the example to the left, getADC3 and getADC2 have been defined as the values for the ADMUX register for the respective ADC input

```
// The readADC function recommended
#define startConversion    ADCSRA |= (1<<ADSC)
#define conversionRunning  ADCSRA & (1<<ADSC)
#define getADC3            0b01100011
#define getADC2            0b01100010

char readADC(char channel)
{
    // Setup the appropriate ADC channel
    ADMUX = channel;
    // Starts a single conversion
    startConversion;
    // Do nothing until the conversion is
    complete
    while(conversionRunning);
    // Return the ADC value
    return(ADCH);
}
```

Setting up VR1

- We will be using a Variable Resistor (VR1)
  - AKA ‘POT1’ on the Lab Board
- Ensure that the selector switch is set to POT1 and not MIC
  - Locate this under the seven-segment displays
  - This switch tells us that this input is shared between the Microphone and Variable Resistor 1



Setting ADC Registers Example – Problem 2

- As we need to switch between two ADC sources, we need to operate in the single-conversion mode
- Setup ADCSRA
  - Enable ADC – Bit 7 = 1
  - We don't want to start a conversion right away – Bit 6 = 0
  - We want to operate in single-conversion mode – Bit 5 = 0
  - We are not using interrupts (for now) – Bit 4 & 3 = 0
  - We will be using the largest pre-scaler (128) – Bit 2:0 = 111
- Setup ADCSRB
  - We don't need to worry about this register, as the default values are correct.

```
// Setup ADC for single-
   conversion mode

void setup(void) {
    ADCSRA = 0b10000111;
}
```

Setting ADC Registers Example – Problem 5

- We have a single ADC source. Hence, we can use the auto-trigger mode to save us calling `readADC()`
- Setup ADCSRA
  - Enable ADC – Bit 7 = 1
  - We want to start the operation of the free-running mode – Bit 6 = 1
  - We want to operate auto-trigger mode – Bit 5 = 1
  - We are not using interrupts (for now) – Bit 4 & 3 = 0
  - We will be using the largest pre-scaler (128) – Bit 2:0 = 111
- Setup ADCSRB
  - We don't need to worry about this register, as the default values are correct.
- Setup ADMUX
  - Since we are running in continuous-conversion mode, setup the ADMUX here!

```
// Setup ADC for auto-
   trigger

void setup(void) {
    ADCSRA = 0b11100111;
    ADCSRA = 0b01100001;
}
```