

MPLAB[®] Code Configurator & Core Independent Peripherals

Lab Manual

This page is intentionally left blank

Lab 2 – FSK Receiver with CIPs

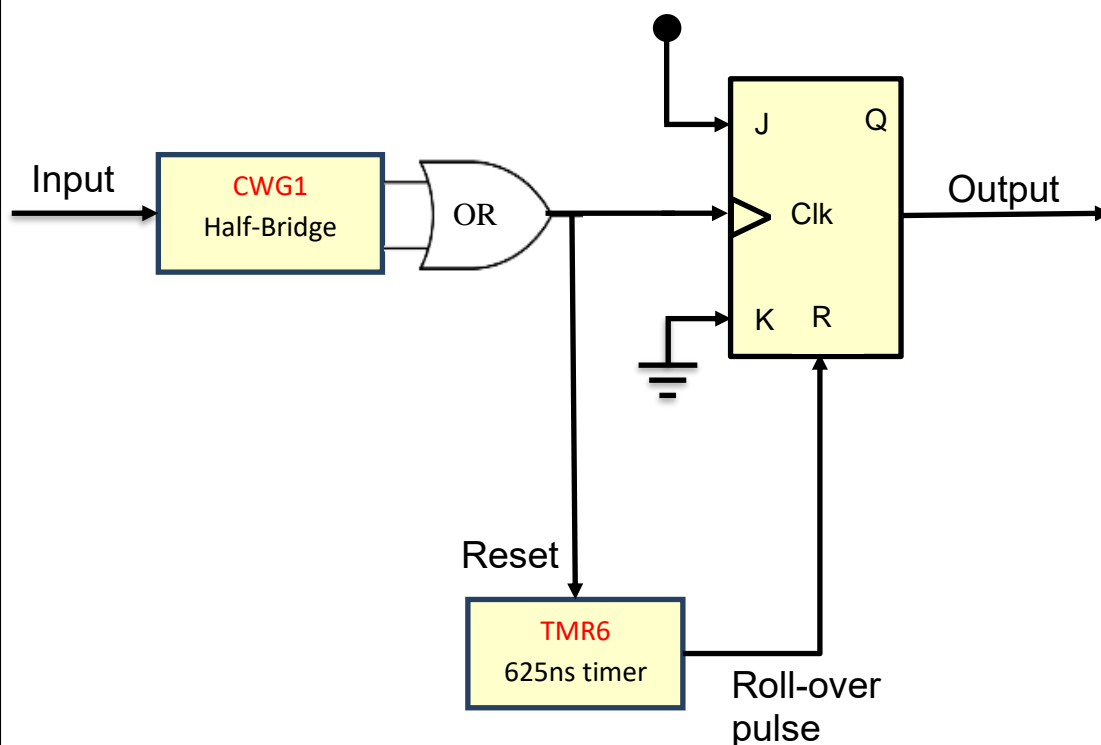
Purpose

This lab demonstrates how to implement FSK receiver by configuring the core-independent peripherals (CIPs) on the PIC16F18446. The CIPs used include Timer, Complementary Waveform Generator (CWG) and Configurable Logic Cell (CLC). CNANO board is used and the project is fully implemented with MPLAB® Code Configurator. The usage of CIPs means that the actual core of the MCU is not involved at all and can either be sleeping to save power or work on other tasks.

Hardware

Function	Pin	Setting
CLC1	-	Input
CWG	-	
CLC2	RC1	Test Point output
CLC3	RC0	De-modulated output
TMR6	-	Roll-over pulse, external reset

Theory of operation



Theory

CLC1 is used to route the input signal from DSM Output to the Complementary Waveform Generator (CWG1). The CWG1 can also use the DSM_OUT directly. CWG1 follows the input signal, but is configured to have a delay of 468.75ns to 500ns before reacting. In simple FSK modulation we have two different input frequencies, in this case 800 kHz & 1.33 MHz. With 800 kHz signal, the input signal changes state every 625ns, so the CWG1 will have time to react to input signal and will thus have output. With 1.33 MHz input signal, the CWG1 won't output anything due to configured delay. So in essence, the CWG1 determines whether we are receiving the higher or lower of the FSK frequencies, and outputs either a stable signal (high freq) or a continuous train of pulses (low freq). CLC2 is used to combine the CWG1A and CWG1B outputs so that we monitor both the high and low signal parts.

In the next step, we use the CWG1 output as an external reset signal to Timer 6. Timer 6 is configured to have a 625ns period, and to output a single pulse when it rolls over (i.e. 625ns has passed without it being reset). This means that when the output of CWG1 is stable (high frequency), the Timer 6 does not get reset, thus it rolls over and outputs a pulse. This output is connected to CLC3, which is configured as a J-K flip-flop with Reset. Timer 6 output is the reset input of the J-K, which means that the output of J-K is set to 0 when Timer 6 rolls over (= when the CWG1 output does not change = when the input signal has high frequency).

In addition to being connected to Timer 6, the CWG1 output is also connected to CLK input of the J-K flip-flop. When the input frequency is low, and the CWG1 output is a train of pulses, these pulses will clock the J-K flip-flop and will set the state of the flip-flop according to its J & K inputs. These are permanently fixed so that J = 1 and K = 0, which will set the output to 1.

Procedure

Connect the CNANO board (DM164144) to the PC.

Depending on your computer, there might be some time needed for driver installation. The debugger on-board uses HID, and the operating system has a driver for it. (Windows, Mac, Linux)

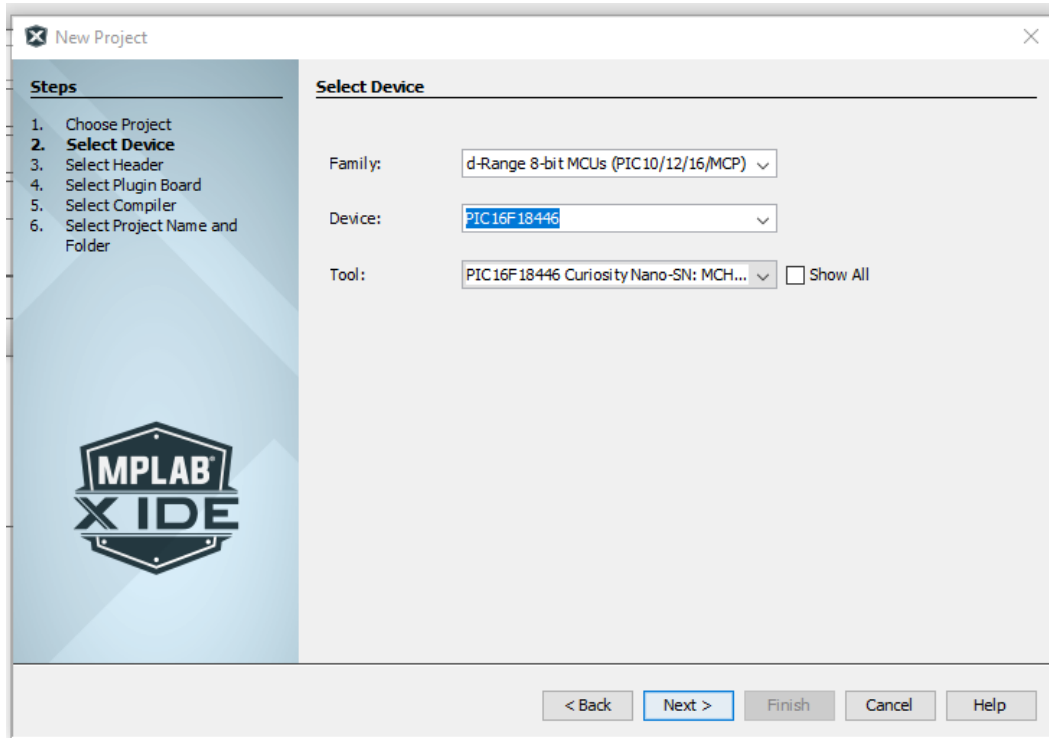
Create a new project using PIC16F18446 on Curiosity Board HPC (screenshots in Lab 1)

Create a new project in the MPLAB® X IDE:

Start MPLAB X, and then go to **File → New Project...**

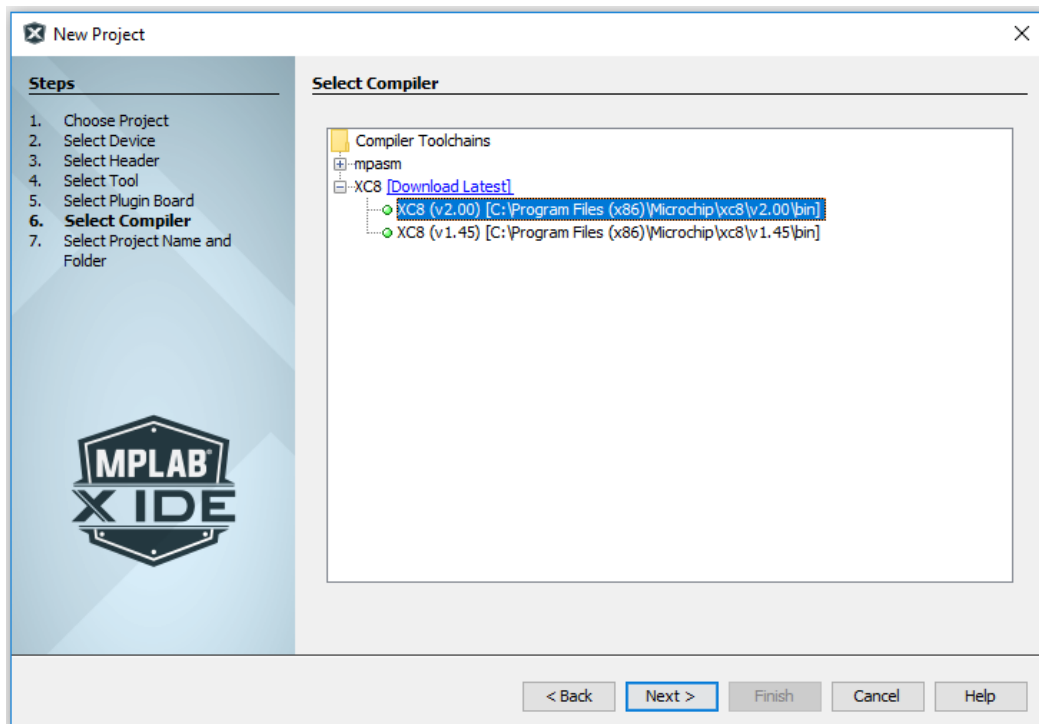
In “New Project” dialog navigate to **Microchip Embedded** Category and select **Standalone Project**

Choose the right HW settings. **Select** from the list the device on the CNANO board (**PIC16F18446**). Please make sure the PIC16F18446 is selected, and **not** the LF variant. From the **Tool** list we will select **Curiosity Nano**.



Choosing the compiler

From the Select Compiler list **select the latest XC8 compiler**. For this lab we use XC8 (v2.20)



Saving the project and finalizing the settings

The **Project Name** for this lab is **FSK_RECEIVER**.

For the **Project Location** we will use **c:\curiosity_ws**

Dismiss the dialog with **Finish**

Start the MPLAB® Code Configurator tool

Select **Tools → Embedded → MPLAB® Code Configurator v4: Open/Close** from the main menu.

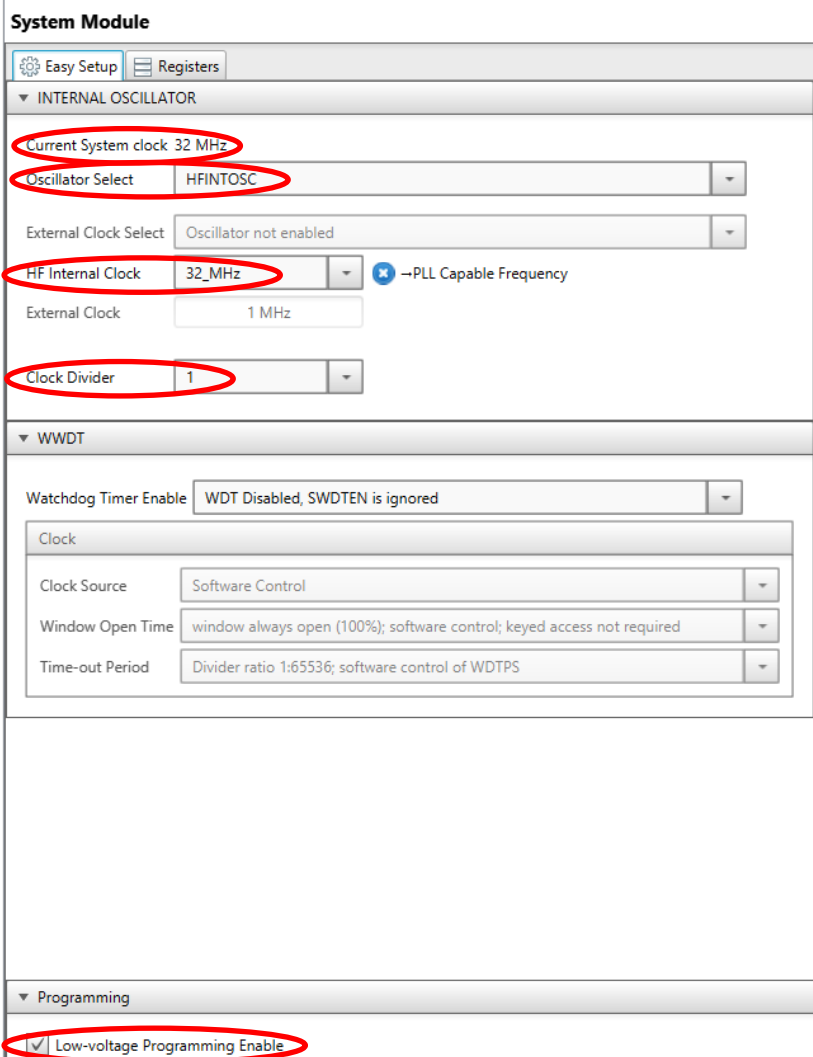
Setup the internal clock and other fuses in the System resource:

From **Project resources** tab, double click on **System Module**.

From the **Oscillator Select** settings, make sure to select **HFINTOSC**

HF Internal Clock should be set to **32_MHz** setting and the **Clock Divider** to **1**. This setup establishes the system clock to 32MHz for the CPU.

As the debugger on board (PKOB) uses **Low Voltage Program** method to program the MCU, ensure that **Low-voltage programming Enable** box is checked:



System Module

Easy Setup Registers

INTERNAL OSCILLATOR

Current System clock 32 MHz

Oscillator Select HFINTOSC

External Clock Select Oscillator not enabled

HF Internal Clock 32_MHz →PLL Capable Frequency

External Clock 1 MHz

Clock Divider 1

WWDTC

Watchdog Timer Enable WDT Disabled, SWDTEN is ignored

Clock

Clock Source Software Control

Window Open Time window always open (100%); software control; keyed access not required

Time-out Period Divider ratio 1:65536; software control of WDTPS

Programming

Low-voltage Programming Enable

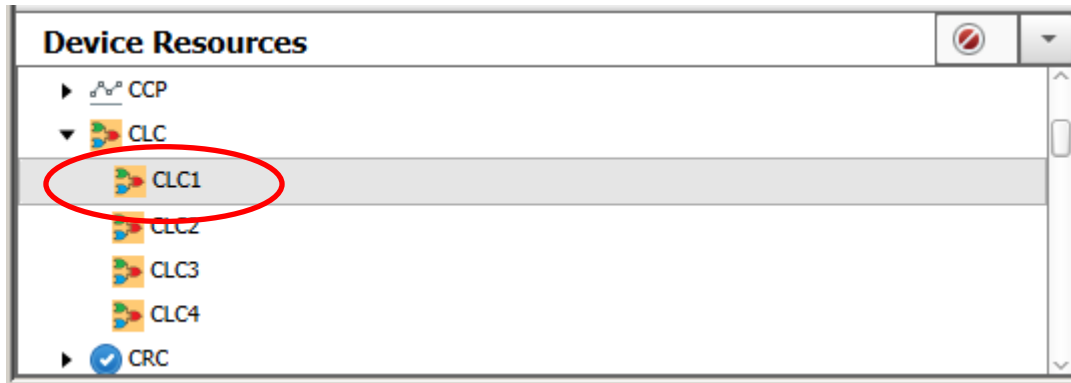
Add the CLC1 driver to the Project Resources

Select the **Resource Management [MCC]** tab.

In the **Device Resources - Peripherals** area scroll-down to the **CLC** entry and expand the list.

(Hint: You may need to click-on the '▶' symbol.)

Now double click-on the **CLC1** entry to add it to the **Project Resources** list.



Configure **CLC1** to route input signal to **CWG**

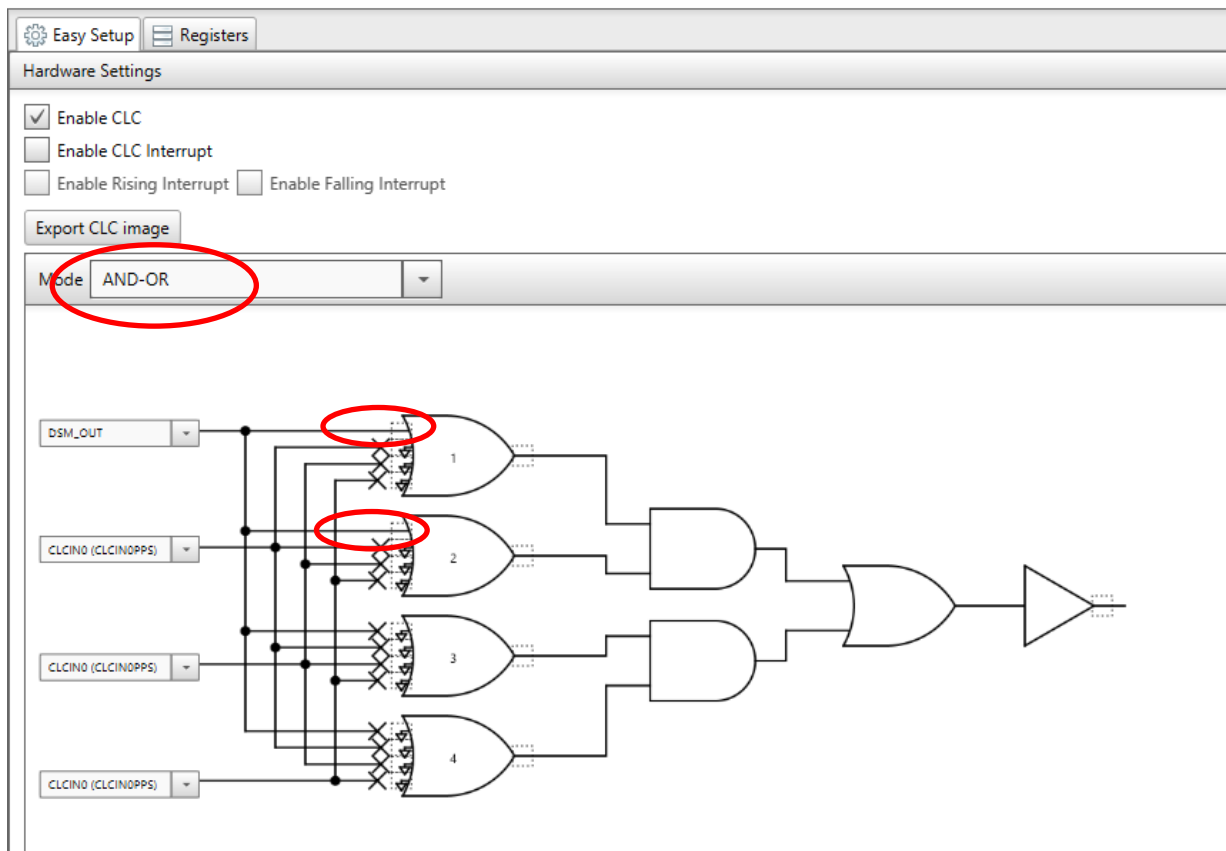
The DSM_OUT (RA1) pin which is the FSK signal on the CNANO board can be directly connected to CWG, but for this lab we will be using CLC1 block to re-route the signal in this step.

Select AND-OR as CLC1 Mode.

Configure one of the inputs as CLCIN0, and route this to OR gates 1 and 2.

Now the AND gate in the next state will have two similar inputs, and we have the input signal available to **CWG**.

CLC1

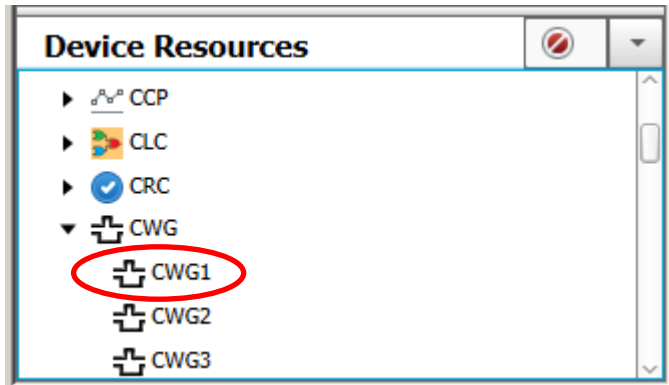


Add the CWG1 to project resources

Select the **Resource Management [MCC]** tab.

In the **Device Resources - Peripherals** area scroll-down to the **CWG** entry and expand the list. (Hint: You may need to click-on the '▶' symbol.)

Now double click-on the **CWG1** entry to add it to the **Project Resources** list.



Configure the CWG

Select **CLC1_OUT** as input source.

Select Output Mode as Half bridge mode.

Set Clock source to **HFINTOSC**.

Expand the 'Events' menu

Set both **Rising Counts** and **Falling Counts** to **15 to 16** to get the desired 468.75-500ns delay for **CWG1**

CWG1

Easy Setup Registers

Hardware Settings

☒ Enable CWG ☐ CWG Interrupt

Input Source: CLC1_OUT

Output Mode: Half bridge mode

Clock Source: HFINTOSC

Dead-band uncertainty: 31.25 ns

▼ Events

Rising Counts: 15 to 16

Minimum Dead-band: 468.75 ns

Maximum Dead-band: 500.0 ns

Falling Counts: 15 to 16

Minimum Dead-band: 468.75 ns

Maximum Dead-band: 500.0 ns

► Output Pin Config

► Auto-Shutdown

Add the CLC2 driver to the Project Resources

Select the **Resource Management [MCC]** tab.

In the **Device Resources - Peripherals** area scroll-down to the **CLC** entry and expand the list.
(Hint: You may need to click-on the '►' symbol.)

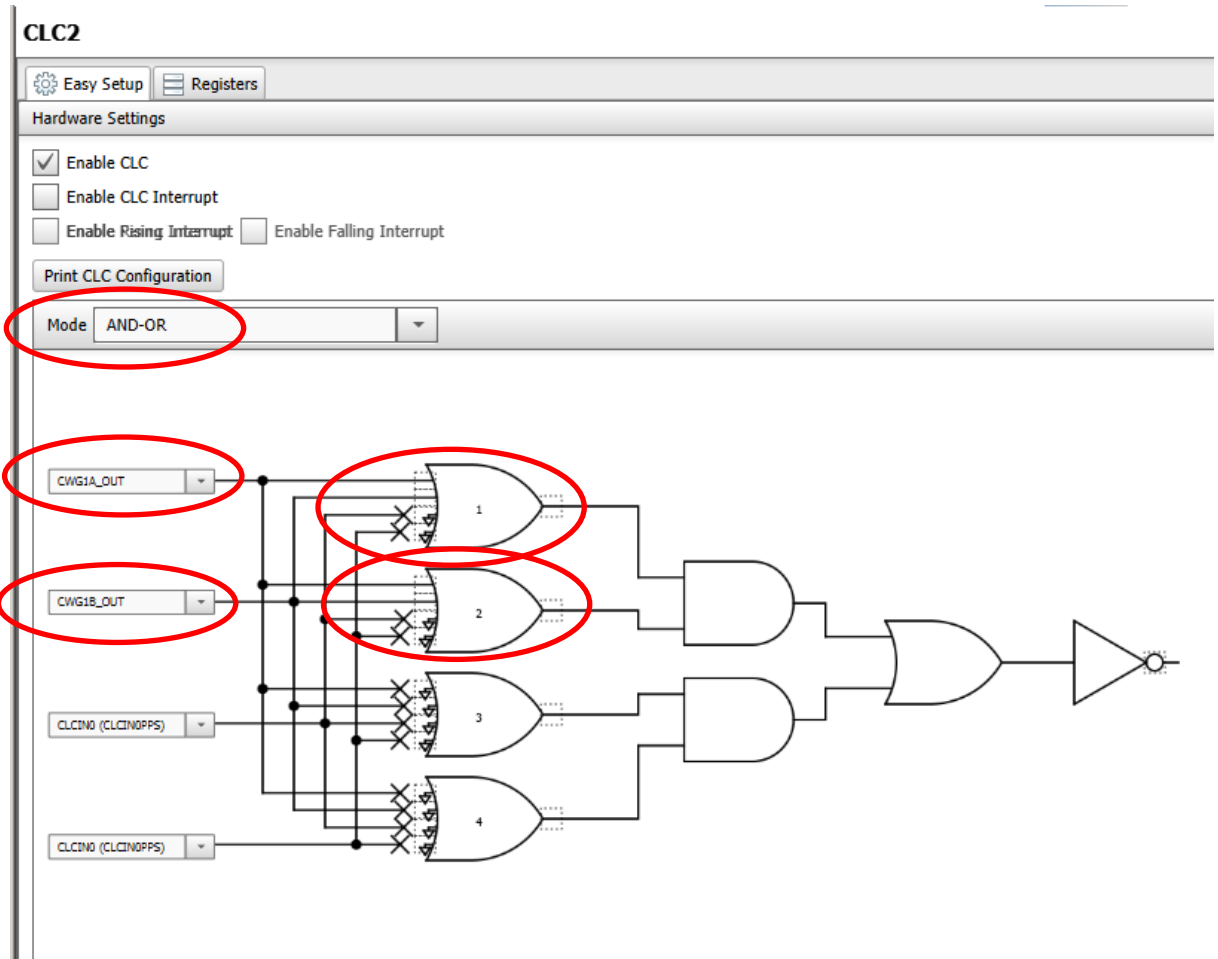
Now double click-on the **CLC1** entry to add it to the **Project Resources** list.

Configure the CLC2

Set the mode to **AND-OR**

Select **CWG1A** as input source 1.

Select **CWG1B** as input source 2.
Connect both inputs to OR gates 1 & 2.



Add the **TMR6** driver to the Project Resources

Select the **Resource Management [MCC]** tab.

In the **Device Resources - Peripherals** area scroll-down to the **TMR** entry and expand the list.
(Hint: You may need to click-on the '►' symbol.)

Now double click-on the **TMR6** entry to add it to the **Project Resources** list.

Configure the **TMR6**

Set the **Clock Source** to **FOSC/4**

Set the **Timer Period**. With 800 kHz & 1.33MHz signals and the selected clock source, 625ns is suitable value.

Select **Ext Reset Source** as **CLC2_out**.

Select **Control Mode** as **Roll over pulse**.

Select **Start/Reset Option** as **Resets at rising/falling TMR6_ers**.

TMR6

Hardware Settings

☒ Enable Timer

Control Mode: Roll over pulse

Ext Reset Source: CLC2_out

Start/Reset Option: Resets at rising/falling TMR6_ers

Timer Clock

Clock Source: FOSC/4

Clock Frequency: 32.768 kHz

Polarity: Rising Edge

Prescaler: 1:1

Postscaler: 1:1

Timer Period

Timer Period: 125 ns ≤ 625 ns ≤ 32 us

Actual Period: 625 ns (Period calculated via Timer Period)

Software Settings

☐ Enable Timer Interrupt

Callback Function Rate: 0x0 x Time Period = 0.0 ns

Add the **CLC3** driver to the Project Resources

Select the **Resource Management [MCC]** tab.

In the **Device Resources - Peripherals** area scroll-down to the **CLC** entry and expand the list.
(Hint: You may need to click-on the '►' symbol.)

Now double click-on the **CLC3** entry to add it to the **Project Resources** list.

Configure the **CLC3**

Set the mode to **JK flip-flop with R**.


Select **CLC2_out** as input source 1.

Select **TMR6** as input source 2.

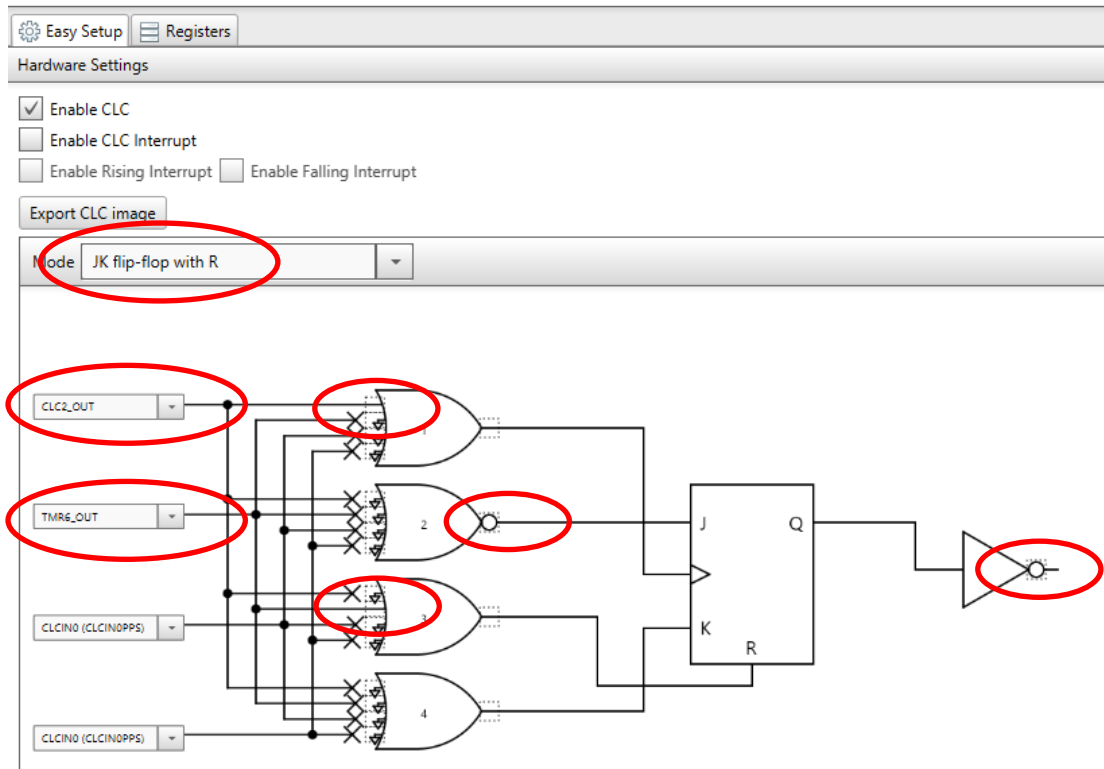
Connect input 1 (CLC2_OUT) to AND gate 1, that goes to CLK of the JK

Connect input 2 (TMR6) to AND gate 3, that goes to R of the JK

The J & K need to be permanently fixed to VDD and GND. To achieve this, we need to know that AND gate output without any input is 0 (GND). So just disable any input to AND gates 2

and 4, and negate the output of AND gate 2. You can change logic connections and straight/negated output by clicking on the dashed squares  on the wires. Invert the output of the JK.

CLC3



Assign the corresponding pins in the **Pin Manager Grid [MCC]** window

To view the output from the FSK demodulator, we can route the output from **CLC3** to some pin, for example **RC0**. This will be TTL serial data at 9600 BAUD which you can connect to a converter to view on a PC.

Note there are other pins selected by default. In this laboratory we are not using those pins and can be deselected or leave them as configured by default.

Pin Manager: Grid View

Package:	QFN20		Pin No:	16	15	14	1	20	19	10	9	8	7	13	12	11	4	3	2	5	6
			Port A ▼				Port B ▼				Port C ▼										
Module	Function	Direction	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7			
CLC1	CLC1OUT	output																			
CLC2	CLC2OUT	output																			
CLC3	CLC3OUT	output																			
CLCx ▼	CLCIN0	input																			
	CLCIN1	input																			
	CLCIN2	input																			
	CLCIN3	input																			
CWG1 ▼	CWG1A	output																			
	CWG1B	output																			
	CWG1C	output																			
	CWG1D	output																			
	CWG1IN	input																			
DSM ▼	DSM1OUT	output																			
	MDCARH	input																			

Configure the Pin module

From **Project resources** tab, click on **Pin Module**.

RC0 and RC1 needs to be configured as digital output, and **RA4** as digital input. Tick or untick the corresponding boxes to ensure this setup. Note that RA4 / 'CLCIN0' is on the list several times due to it being a default input to CLC2 and CLC3 as well, but since it's not used in those blocks, we can ignore these lines.

Pin Module

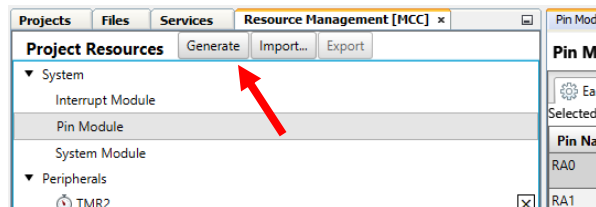
Easy Setup | Registers

Selected Package: QFN20

Pin Name	Module	Function	Custom Name	Start High	Analog	Output	WPU	OD	IOC
RA0	DSM	DSM1OUT		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RA1	DSM	DSM1OUT		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RA2	Pin Module	GPIO	LED_D2	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RA4	CLC1	CLCIN0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RA4	CLC2	CLCIN0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RA4	CLC3	CLCIN0		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RC0	CLC3	CLC3OUT		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RC1	CLC2	CLC2OUT		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RC2	Pin Module	GPIO	Button_S1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	none
RC3	CLC1	CLC1OUT		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none
RC6	EUSART1	TX1		<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	none

Generate the configuration code for the design

Click-on the **Generate** button in the **Resource Management [MCC]** tab.



Make/build the design and program the target device

Select the **Make and Program Device** button or select **Run → Run Main Project** from the main menu.



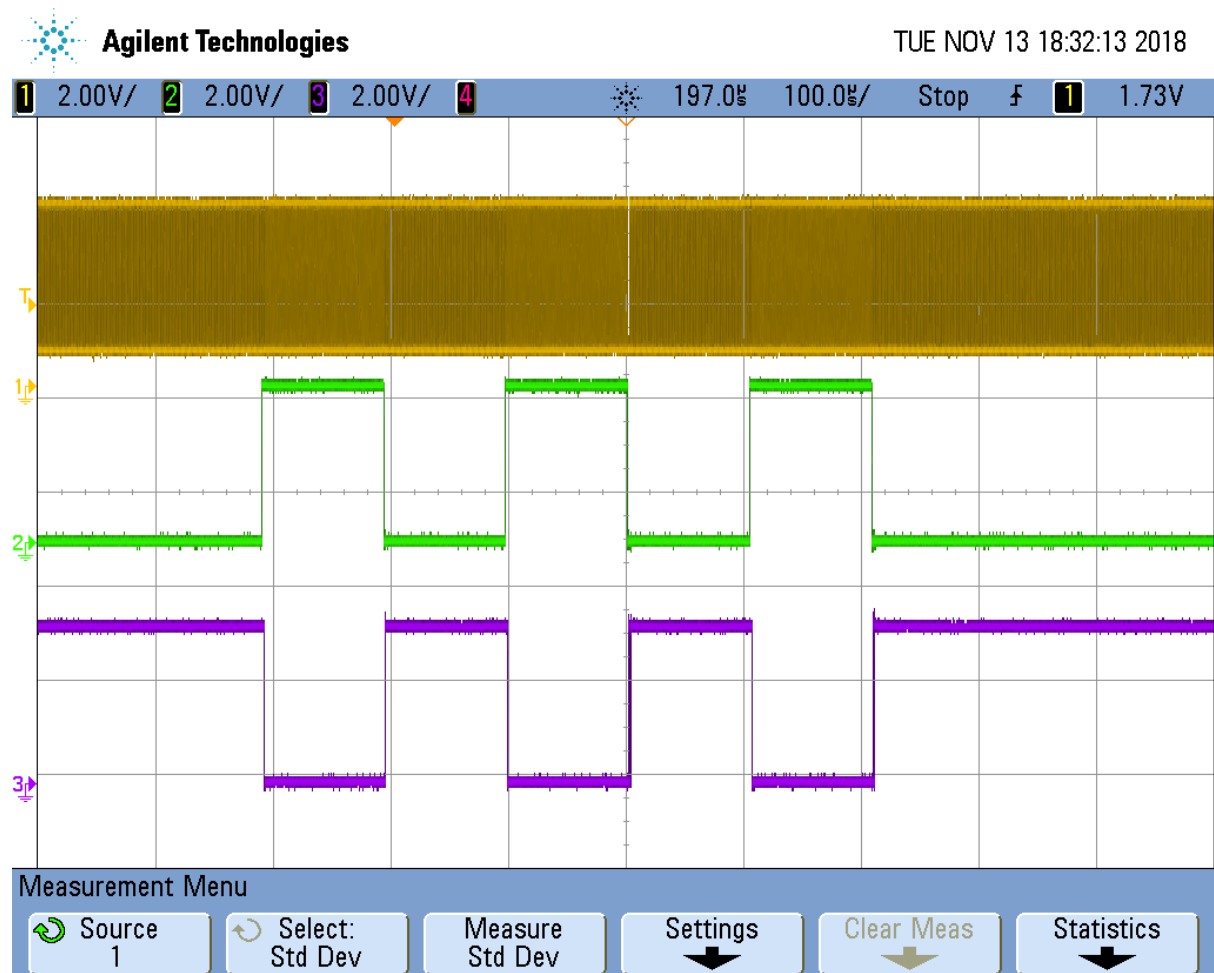
Note: If you receive a warning message about the selection of a 5V programmable voltage, simply click on the **OK** button.

Results

The FSK demodulator is now ready, and any FSK input to RA4 pin (with frequencies of 800 kHz and 1.33 MHz) will be demodulated and output to pin RC0.

To try this out in practice, you can combine the FSK RX & TX projects. This is easiest if you take the TX project as a starting point, and add the peripherals as described here in this lab manual. Remember to also change the pin configuration (RA4 as input, RC0 as output). Then you can observe the demodulated signal on the RC0 pin, and see it change when you press the S1 button. You can

also use a serial terminal like PuTTY to monitor the data. The data is what the FSK transmitter is sending from its lab.



Yellow = FSK signal

Green = UART signal modulating the FSK

Purple = demodulated UART (inverted)

Note