MICROCHIP

Peripheral Trigger Generator

Goals

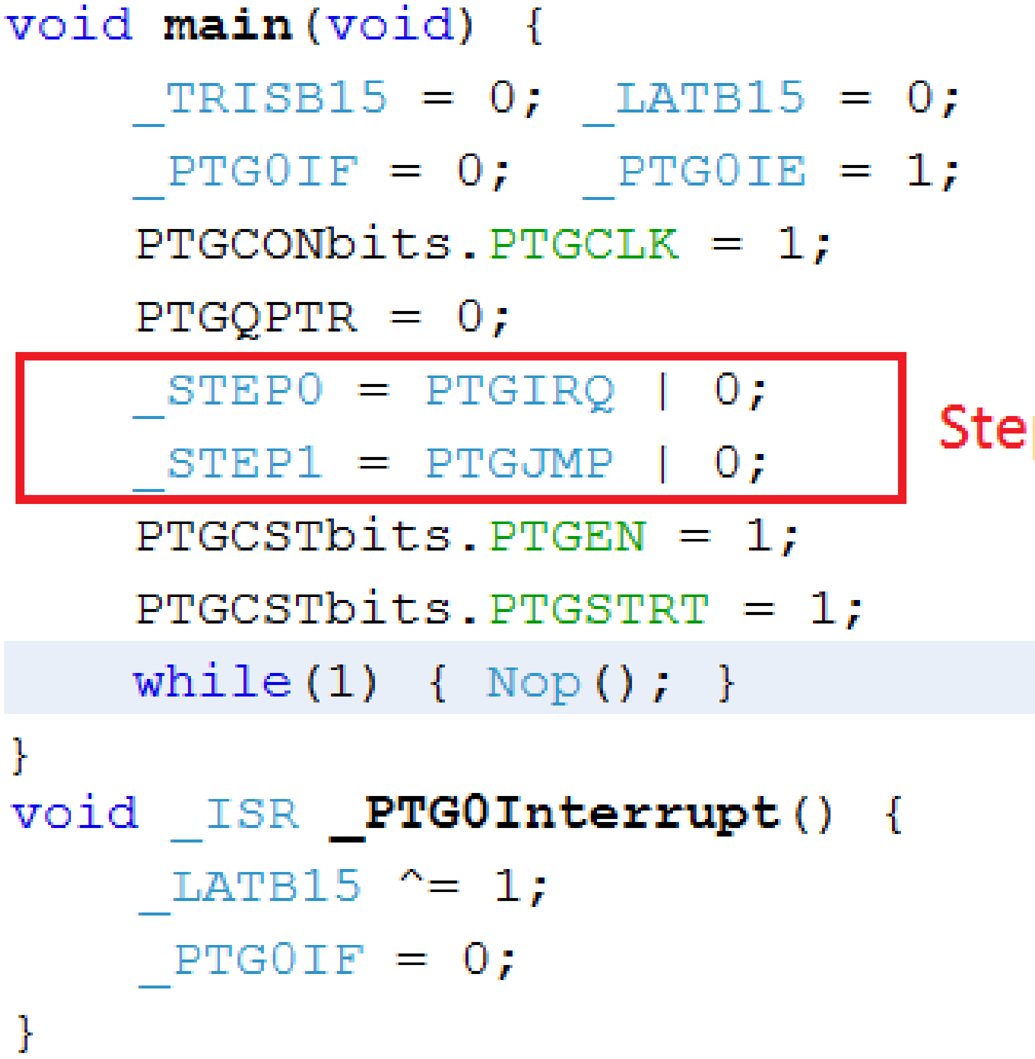
- What is the PTG?
- Why use it?
- How is it used?
- What can it do?
- Where can I learn more?

What is the PTG?

- **The PTG is a programmable step sequencer…**

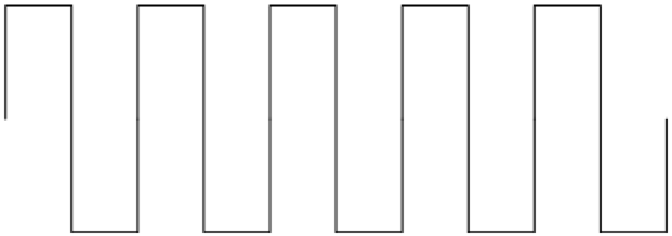
dsPIC/PIC	PTG
Program Memory (Flash)	Step Queue (Ram)
Instruction Set	<Command,Option> pairs form 8-bit Instructions
Operates on data	Operates on triggers and Delays
Program Counter	Queue Pointer (QPTR)
General Purpose RAM	Limit Register
Configuration Bits	Configuration SFRs

Step Queue



```
void main(void) {
    _TRISB15 = 0;  _LATB15 = 0;
    _PTG0IF = 0;   _PTG0IE = 1;
    PTGCONbits.PTGCLK = 1;
    PTGQPTR = 0;
    _STEP0 = PTGIRQ | 0;
    _STEP1 = PTGJMP | 0;
    PTGCSTbits.PTGEN = 1;
    PTGCSTbits.PTGSTRT = 1;
    while(1) { Nop(); }
}
void _ISR _PTG0Interrupt() {
    _LATB15 ^= 1;
    _PTG0IF = 0;
}
```

Step Queue



LATB15 Waveform

Command Categories

Output	Input	Branching	Control
PTGIRQ	PTGWHI	PTGJMP	PTGCTRL
PTGTRIG	PTGWLO	PTGJMPC0	PTGADD
PTGSTRB		PTGJMPC1	PTGCOPY

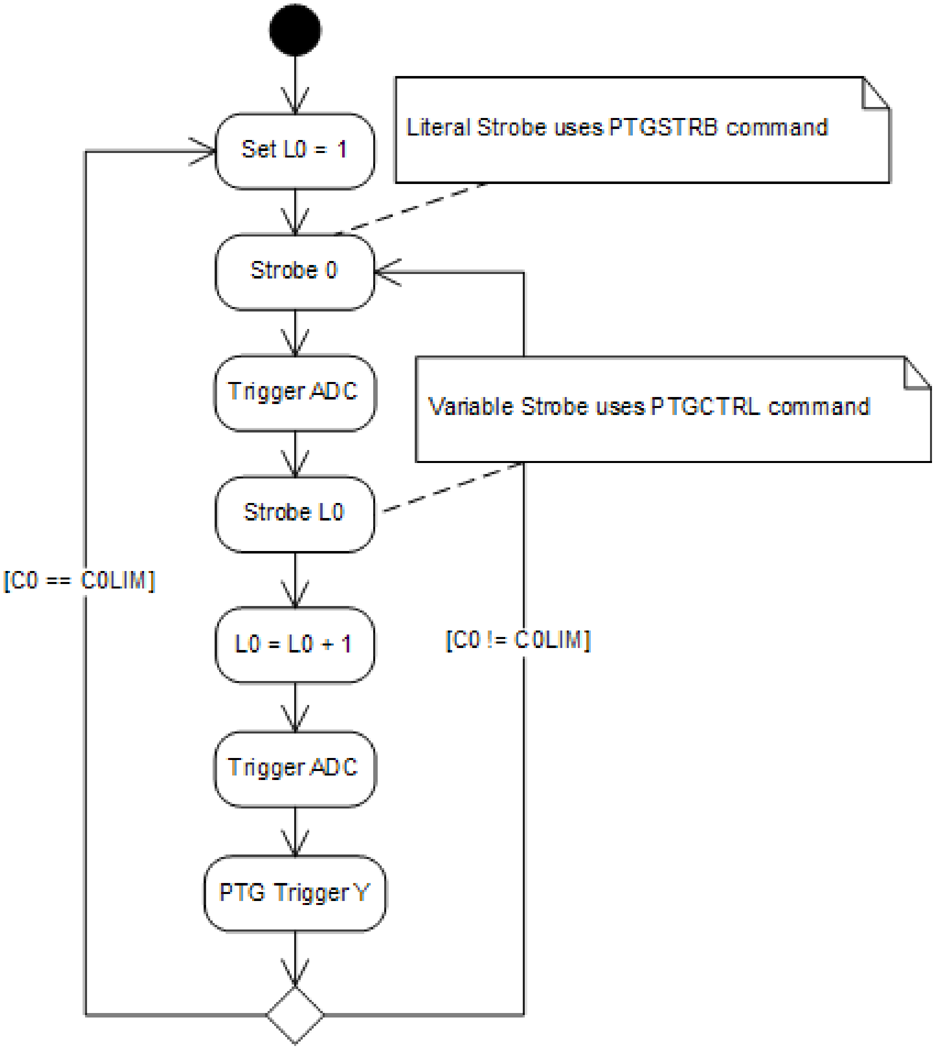
Output Commands

- **PTGTRIG (dsPIC33EPXXXGP50X)**

PTG Output Number	Description
PTGO0	Trigger/Synchronization Source for OC1
PTGO4	Clock Source for OC1
PTGO8	Trigger/Synchronization Source for IC1
PTGO12	Sample Trigger for ADC
PTGO16	PWM Time Base Synchronous Source for PWM
PTGO18	Mask Input Select for Op Amp/Comparator
PTGO30	PTG Output to PPS Input Selection

Output Commands

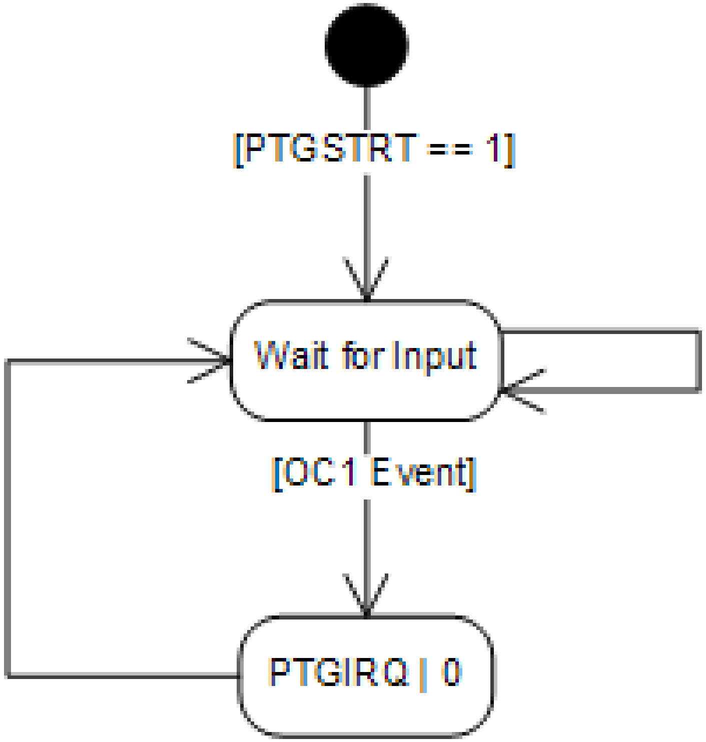
- **PTGSTRB**

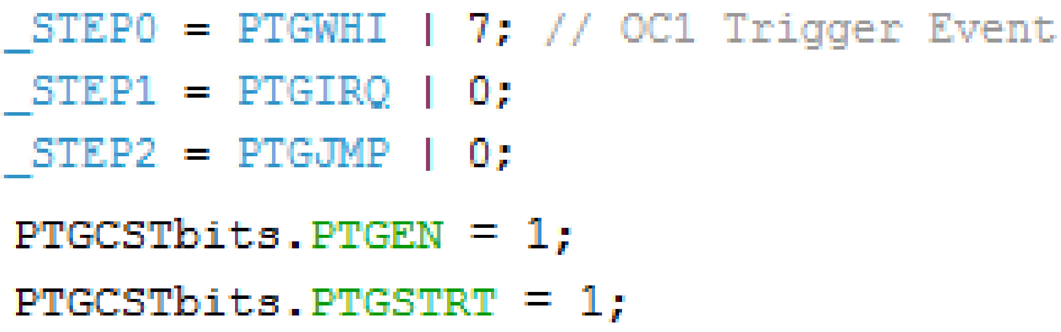


Input Commands

- **PTGWHI / PTGWLO**
- PWM Special event Trigger
- PWM Master time base synchronization output
- PWM[1,3] Interrupt
- OC[1,2] Trigger Event
- IC1 Trigger Event
- CMP[1,4]Trigger Event
- ADC conversion done interrupt
- INT2 external Interrupt

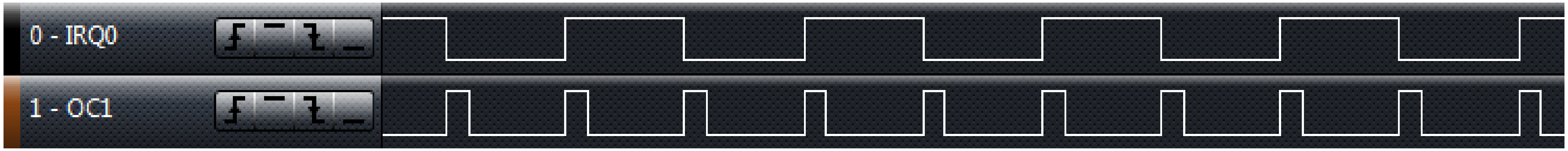
Input Commands





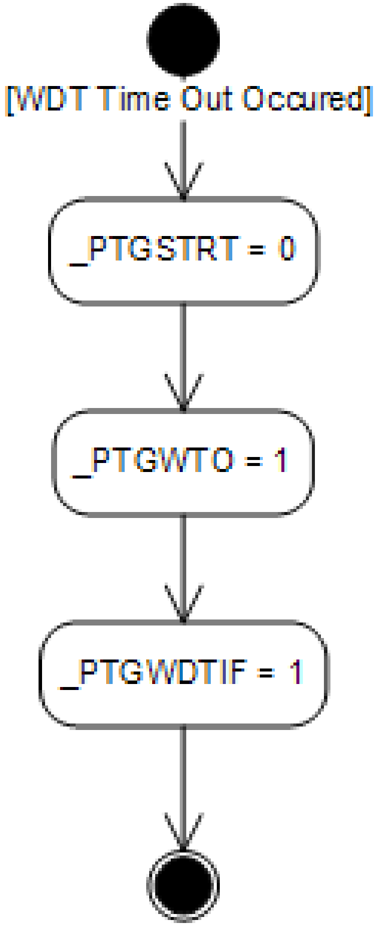
```
_STEP0 = PTGWHI | 7; // OC1 Trigger Event
_STEP1 = PTGIRQ | 0;
_STEP2 = PTGJMP | 0;

PTGCSTbits.PTGEN = 1;
PTGCSTbits.PTGSTRT = 1;
```

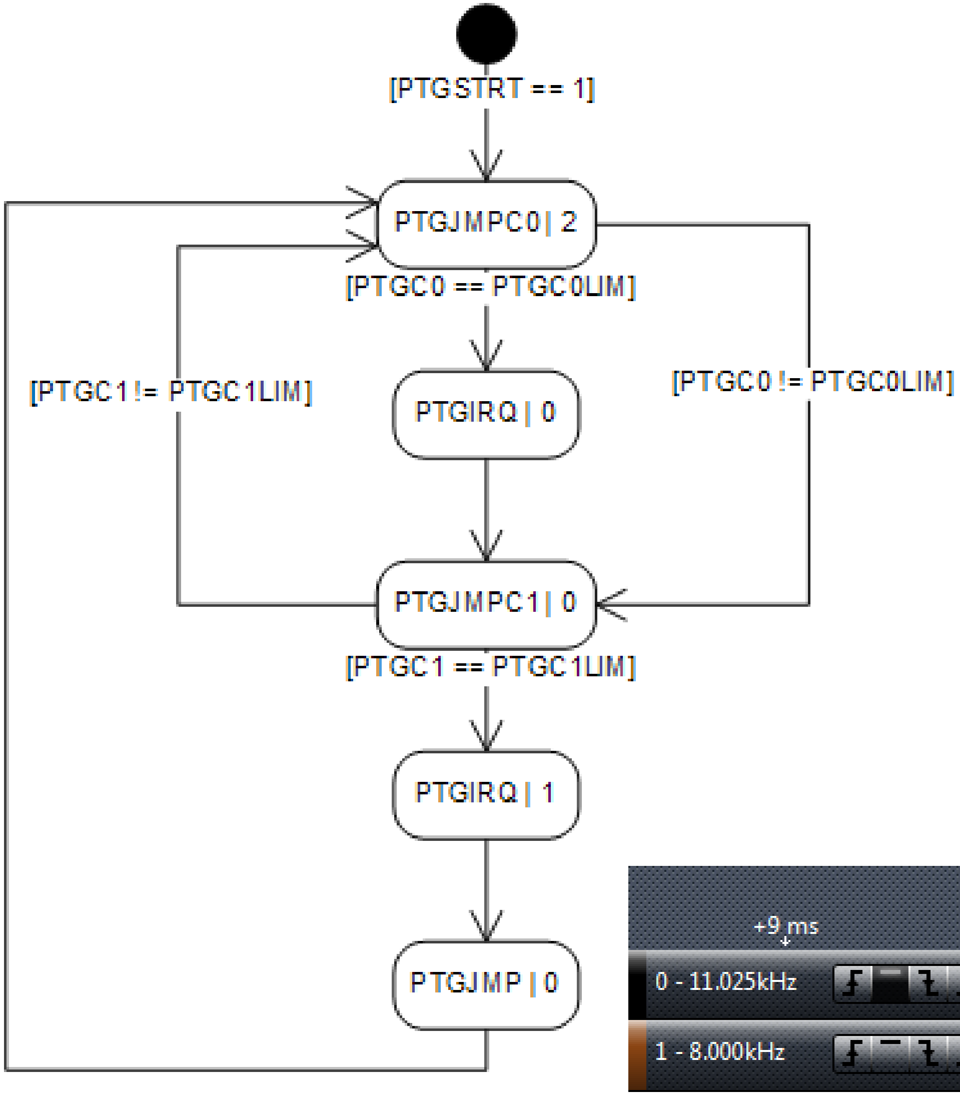


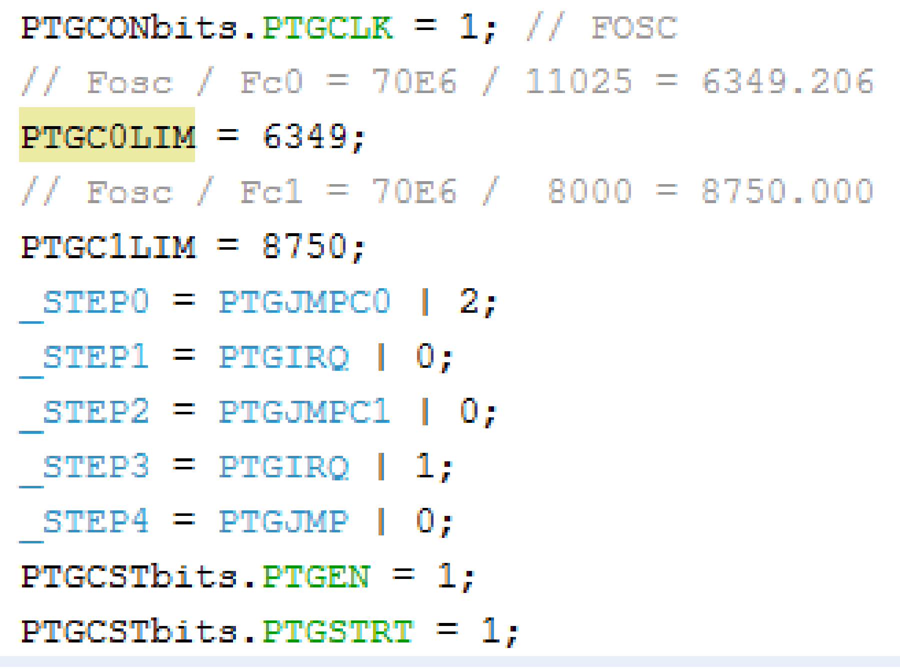
PTG Watchdog Timer

- **What if the input doesn't come when expected?**

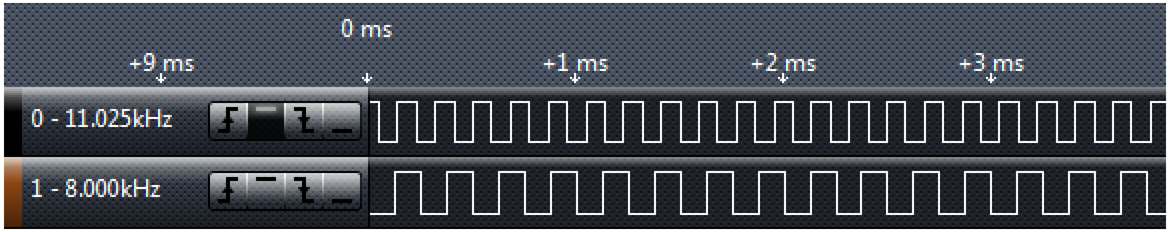


Looping





```
PTGCONbits.PTGCLK = 1; // FOSC
// Fosc / Fc0 = 70E6 / 11025 = 6349.206
PTGC0LIM = 6349;
// Fosc / Fc1 = 70E6 /  8000 = 8750.000
PTGC1LIM = 8750;
_STEP0 = PTGJMPC0 | 2;
_STEP1 = PTGIRQ | 0;
_STEP2 = PTGJMPC1 | 0;
_STEP3 = PTGIRQ | 1;
_STEP4 = PTGJMP | 0;
PTGCSTbits.PTGEN = 1;
PTGCSTbits.PTGSTRT = 1;
```

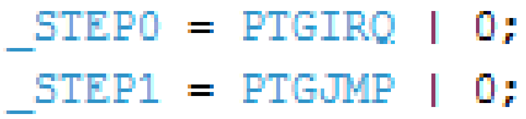


Control Commands

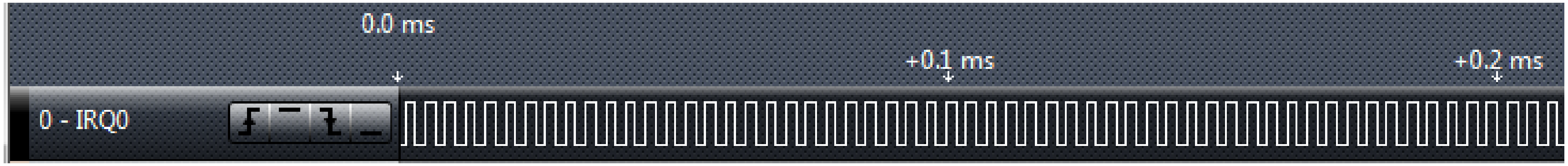
- **PTGCTRL**
- Enable/Disable Step Delay Timer
- Start and wait for the PTG Timer[0,1] to match the PTGT[0,1]LIM
- Wait for the software Trigger (Edge or Level sensitive)
- Copy the PTGC[0,1]LIM register to the AD1CHS0 register
- Copy the PTGL0 register contents to the AD1CHS0 register
- Broadcast Trigger

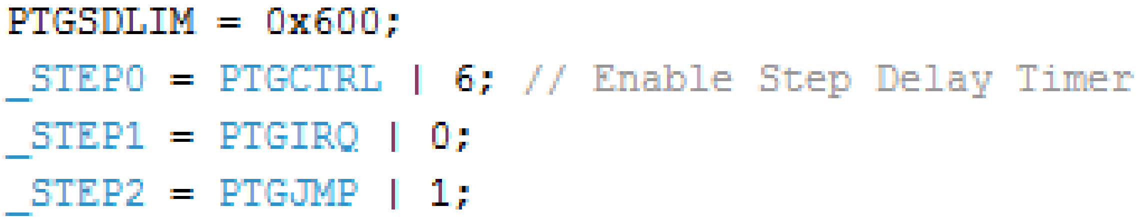
Control Commands

- **PTGCTRL: Step Delay Timer**

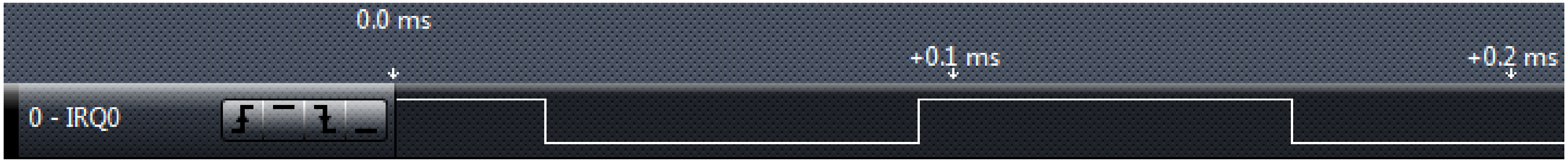


```
_STEP0 = PTGIRQ | 0;
_STEP1 = PTGJMP | 0;
```





```
PTGSDLIM = 0x600;
_STEP0 = PTGCTRL | 6; // Enable Step Delay Timer
_STEP1 = PTGIRQ | 0;
_STEP2 = PTGJMP | 1;
```

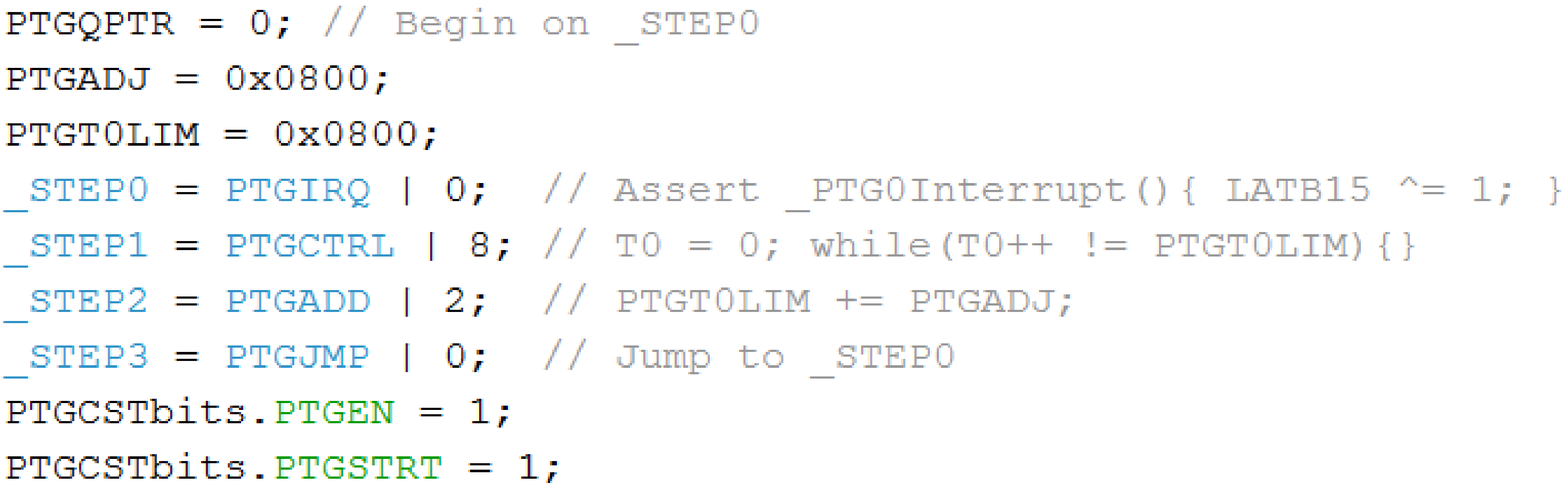


Control Commands

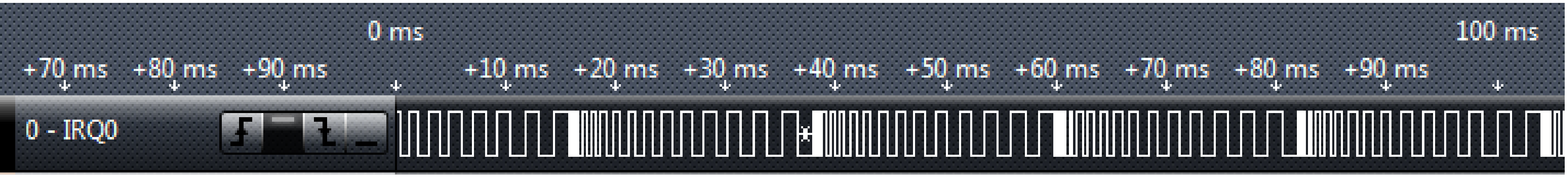
- **PTGADD**
  - PTGC0LIM += PTGADJ
  - PTGC1LIM += PTGADJ
  - PTGT0LIM += PTGADJ
  - PTGT1LIM += PTGADJ
  - PTGSDLIM += PTGADJ
  - PTGL0 += PTGADJ
- **PTGCOPY**
  - PTGC0LIM = PTGHOLD
  - PTGC1LIM = PTGHOLD
  - PTGT0LIM = PTGHOLD
  - PTGT1LIM = PTGHOLD
  - PTGSDLIM = PTGHOLD
  - PTGL0 = PTGHOLD

Control Commands

- **PTGCTRL: Wait for Timer**



```
PTGQPTR = 0; // Begin on _STEP0
PTGADJ = 0x0800;
PTGT0LIM = 0x0800;
_STEP0 = PTGIRQ | 0;  // Assert _PTG0Interrupt(){ LATB15 ^= 1; }
_STEP1 = PTGCTRL | 8; // T0 = 0; while(T0++ != PTGT0LIM){}
_STEP2 = PTGADD | 2;  // PTGT0LIM += PTGADJ;
_STEP3 = PTGJMP | 0;  // Jump to _STEP0
PTGCSTbits.PTGEN = 1;
PTGCSTbits.PTGSTRT = 1;
```



Debugging Step Programs

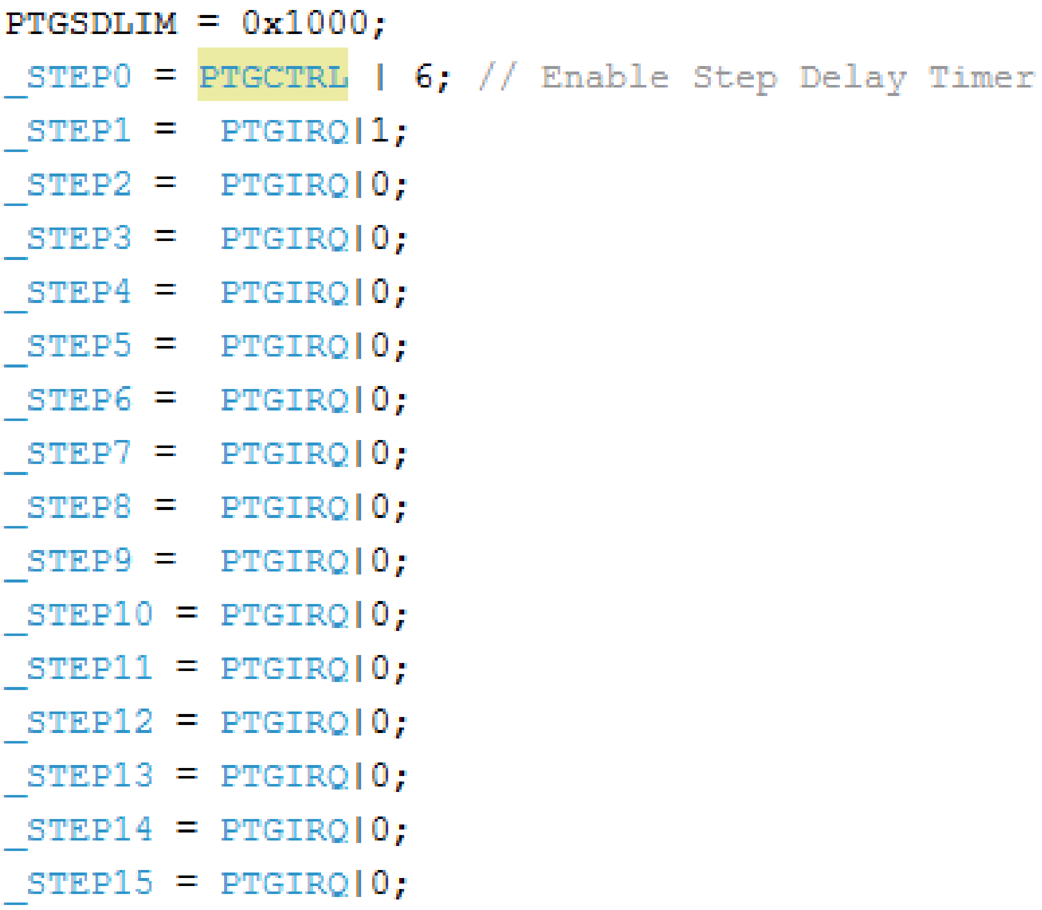
- **PTCST<PTGSSEN> – Enable single stepping.**
- **PTCST<PTGIVIS> – Makes internal counters and timers visible.**
- **When single stepping is enabled, set PTGSTRT to execute the next Step Command.**
- **Step Interrupt ISR Available**

PTG Clock

- **FOSC or FCY**
- **T1/T2/T3 clock**
- **ADC clock**
  - Clock divisor
  - WDT counts PTG clock
- **Clock plays an important role**
  - Delay between PTG enable and start of command execution: 6 x PTG clock period (max)
  - Delay between PTGEN=0 and the complete disabling of the module: 2 x PTG clock period (max)

What happens when…

- **The end of the Step Queue is reached?**



```
PTGSDLIM = 0x1000;
_STEP0 = PTGCTRL | 6; // Enable Step Delay Timer
_STEP1 =  PTGIRQ|1;
_STEP2 =  PTGIRQ|0;
_STEP3 =  PTGIRQ|0;
_STEP4 =  PTGIRQ|0;
_STEP5 =  PTGIRQ|0;
_STEP6 =  PTGIRQ|0;
_STEP7 =  PTGIRQ|0;
_STEP8 =  PTGIRQ|0;
_STEP9 =  PTGIRQ|0;
_STEP10 = PTGIRQ|0;
_STEP11 = PTGIRQ|0;
_STEP12 = PTGIRQ|0;
_STEP13 = PTGIRQ|0;
_STEP14 = PTGIRQ|0;
_STEP15 = PTGIRQ|0;
```

Resources

- Code Examples
- FRM
- Datasheets