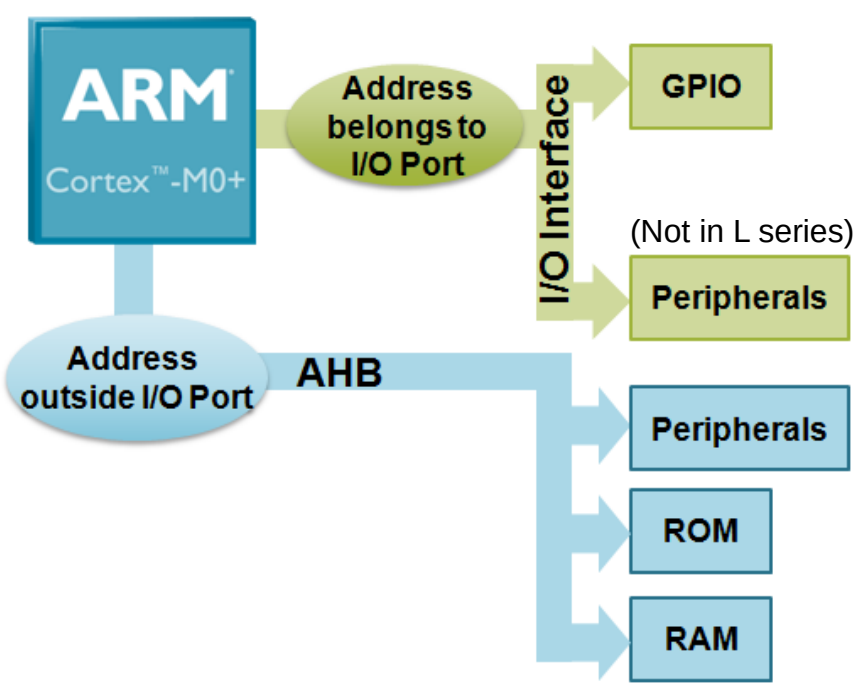
KINETIS MO+ GPIO

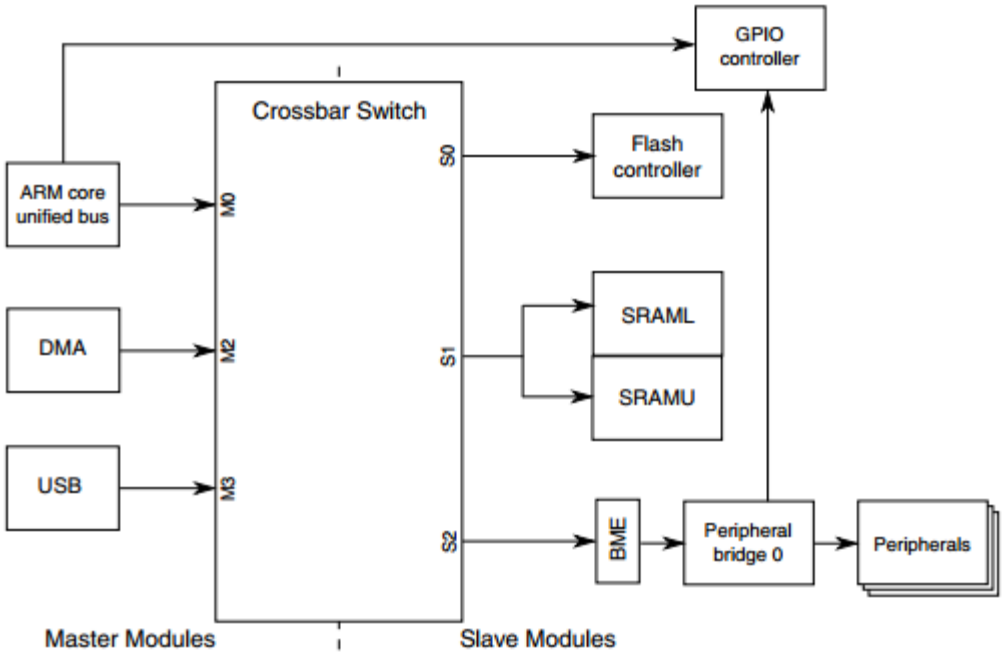
NORMAL GPIO VS. FAST GPIO

DAVID CHEN | SR. FAE

GPIO on Kinetis Cortex-M0+ series

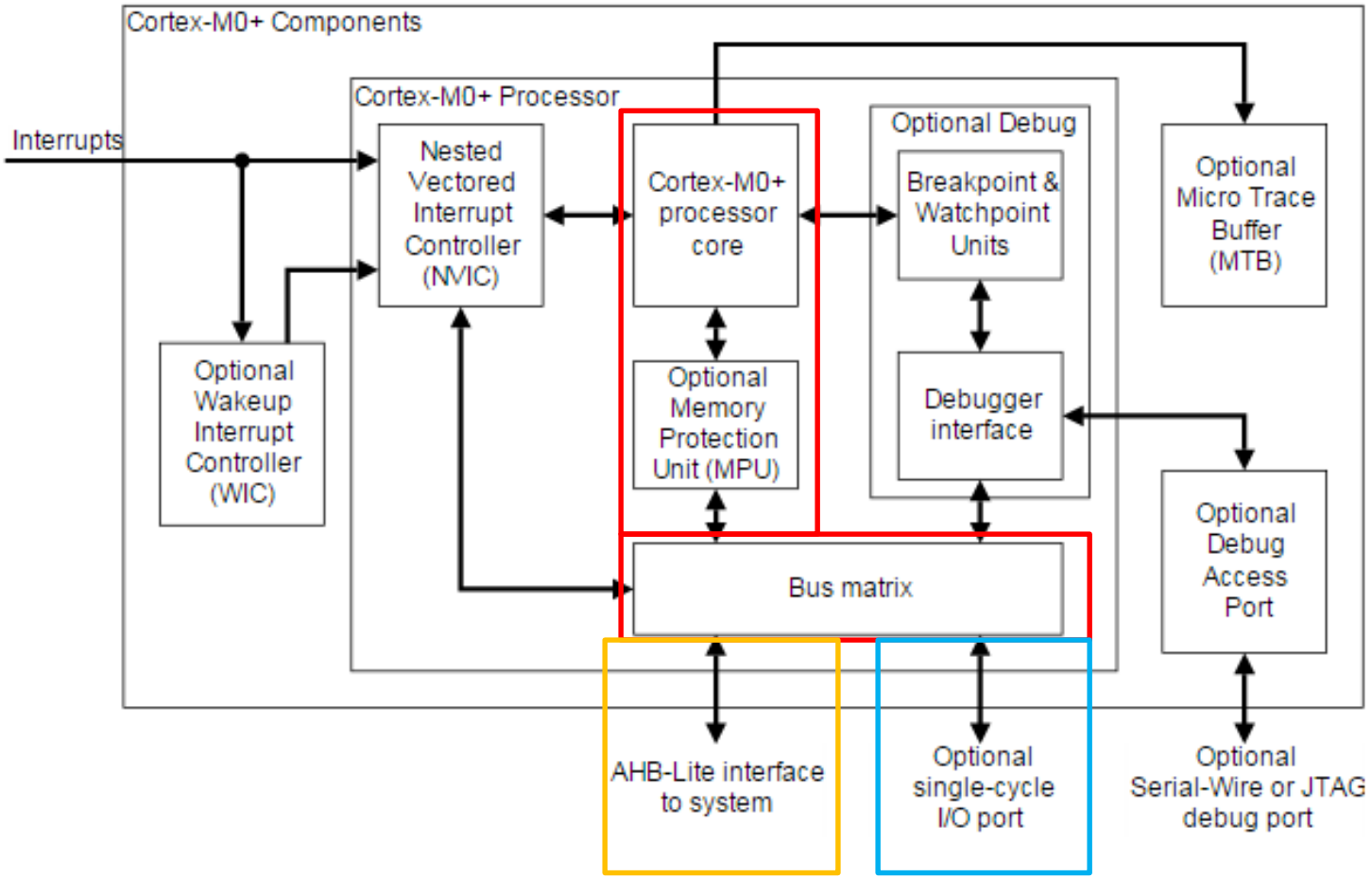
- There are two kinds GPIO control .
  1. Normal Control : Using Peripheral Bridge control GPIO Controller .
  2. single-cycle I/O / Fast GPIO : Using Cortex-M0+ core control GPIO controller .





Cortex-M0+ implementation

- The Cortex-M0+ processor implements a dedicated single-cycle I/O port for high-speed, single-cycle access to peripherals. The single-cycle I/O port is memory mapped and supports all the load and store instructions. The single-cycle I/O port does not support code execution.



Single-cycle IO

- Single-cycle access to I/O: Up to 50 percent faster than standard I/O, improves reaction time to external events allowing bit banging and software protocol emulation .
- Zero wait state access to GPIO registers through IOPORT
  - The GPIO registers are also aliased to the IOPORT interface on the Cortex-M0+ from address $F800_0000. Accesses via the IOPORT interface occur in parallel with any instruction fetches and will therefore complete in a single cycle.

Experiments – GPIO Access Speed

- Purpose : Verify GPIO access speed with different control mode
- Tool requirement :
  - CodeWarrior Ver 10.6
  - FRDM-KL25Z
  - Oscilloscope
- Method :
  - Using different control mode toggle PTD1 then measure time of IO state change . The time is what GPIO control instructions needed .
  - (運用兩種不同的控制方式Toggle PTD1 , 使用示波器量測IO變化時間 ， 也就是執行指令所需時間 .)
    - 1. Through Peripheral Bridge control GPIO
    - 2. Single Cycle IO (Fast IO)

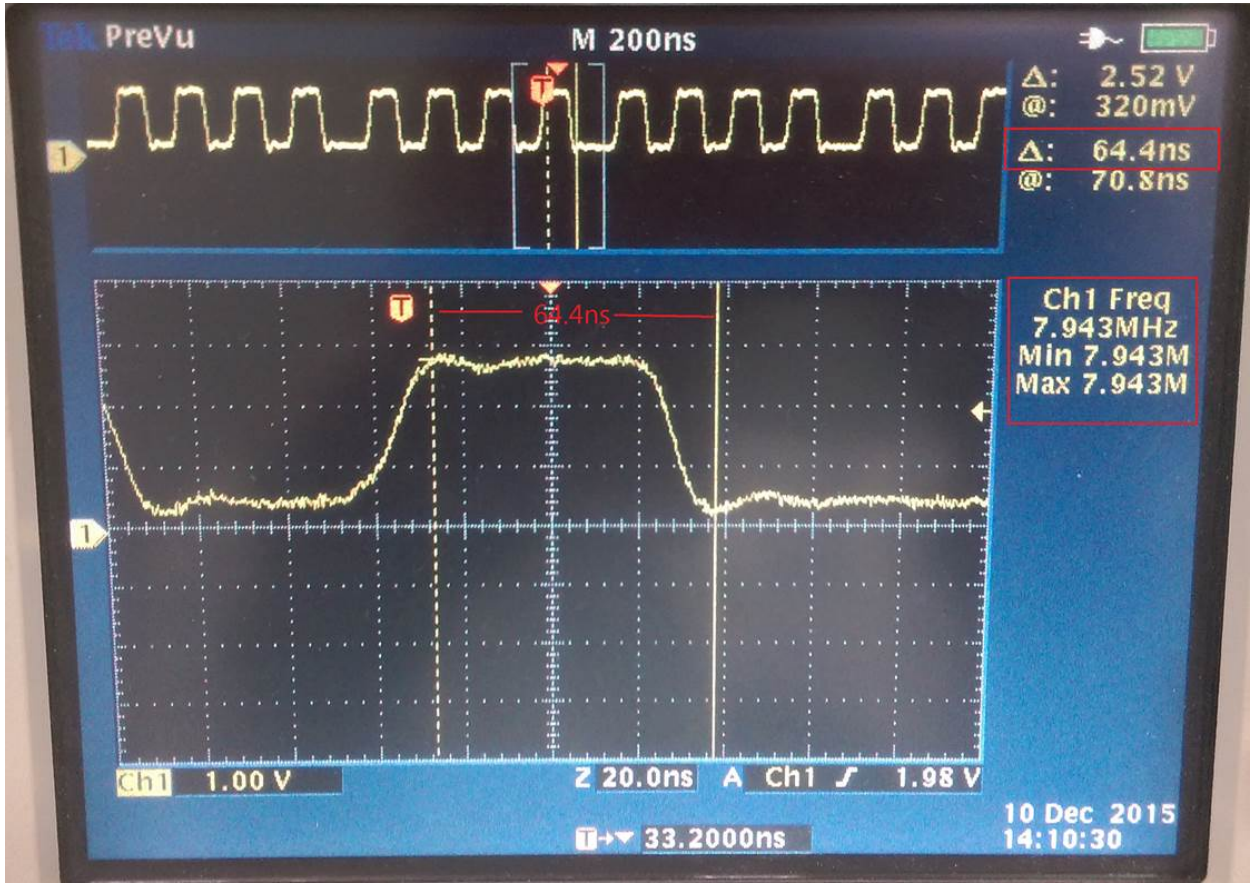
Through Peripheral Bridge control GPIO

- The following code is used to toggle PTD1 .
- The register write only takes one assembly instruction .

```
PTD_BASE_PTR->PCOR = 0x02;
                                        /***** Disassemble
**********************
                                        ldr r2, [pc, #24] ; (20 <main+0x20>)
                                        movs      r3, #2
                                        str r3, [r2, #8]
                                        ***************************************/
  while(1)
 {
     PTD_BASE_PTR->PSOR = 0x02;
                                        /* str r3, [r2, #4] */      //only one instruction
     PTD_BASE_PTR->PCOR = 0x02;
                                        /* str r3, [r2, #8] */
     PTD_BASE_PTR->PTOR = 0x02;
                                        /* str r3, [r2, #12] */
     GPIOD_PTOR = 0x02 ;
                                        /* str r3, [r2, #12] */
     GPIOD_PTOR = 0x02 ;
                                        /* str r3, [r2, #12] */
                                        /*
                                        b.n    c <main+0xc>
                                        nop                ; (mov r8, r8)
                                        .word  0x400ff0c0
                                        */
}
```

Through Peripheral Bridge control GPIO - Result

- The GPIO state change from high to low that took 64.4n sec in one instruction.
- The core clock is 48MHz (20.8ns) , that mean almost three instruction cycle required for once GPIO access .



Single Cycle IO (Fast IO)

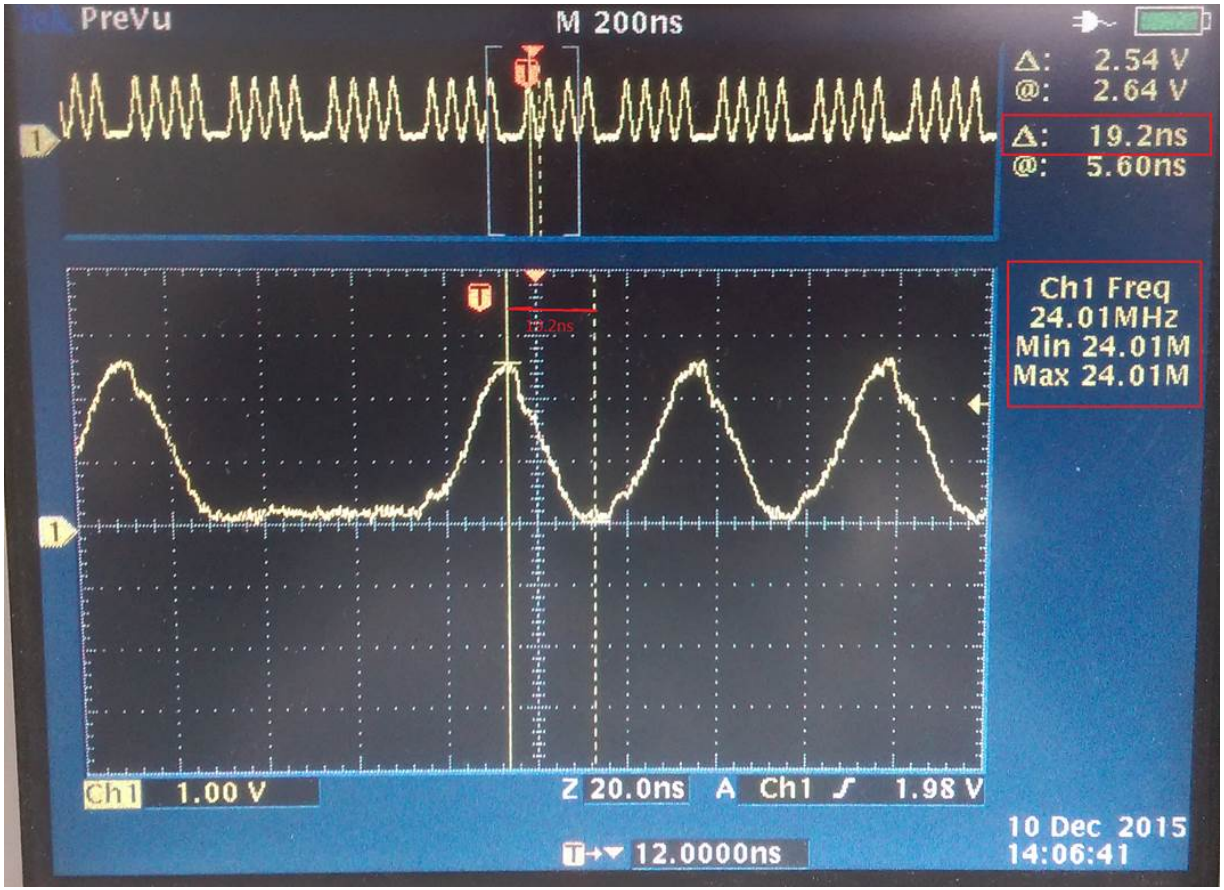
- The following code is used to toggle PTD1 .
- The register write only takes one assembly instruction .

```
FPTD_BASE_PTR->PCOR = 0x02;

while(1)
  {
       FPTD_BASE_PTR->PSOR = 0x02;

       FPTD_BASE_PTR->PCOR = 0x02;

       FPTD_BASE_PTR->PTOR = 0x02;

              FGPIOD_PTOR = 0x02 ;

              FGPIOD_PTOR = 0x02 ;

  }
```

```
/***** Disassemble ***********************
*ldr   r2, [pc, #24] ; (20 <main+0x20>)
*movs  r3, #2
*str   r3, [r2, #8]
****************************************/

     /* str r3, [r2, #4] */

     /* str r3, [r2, #8] */

     /* str r3, [r2, #12] */

     /* str r3, [r2, #12] */

     /* str r3, [r2, #12] */

     /*
      b.n   c <main+0xc>
      nop                   ; (mov r8, r8)
      .word 0xf80ff0c0
     */
```

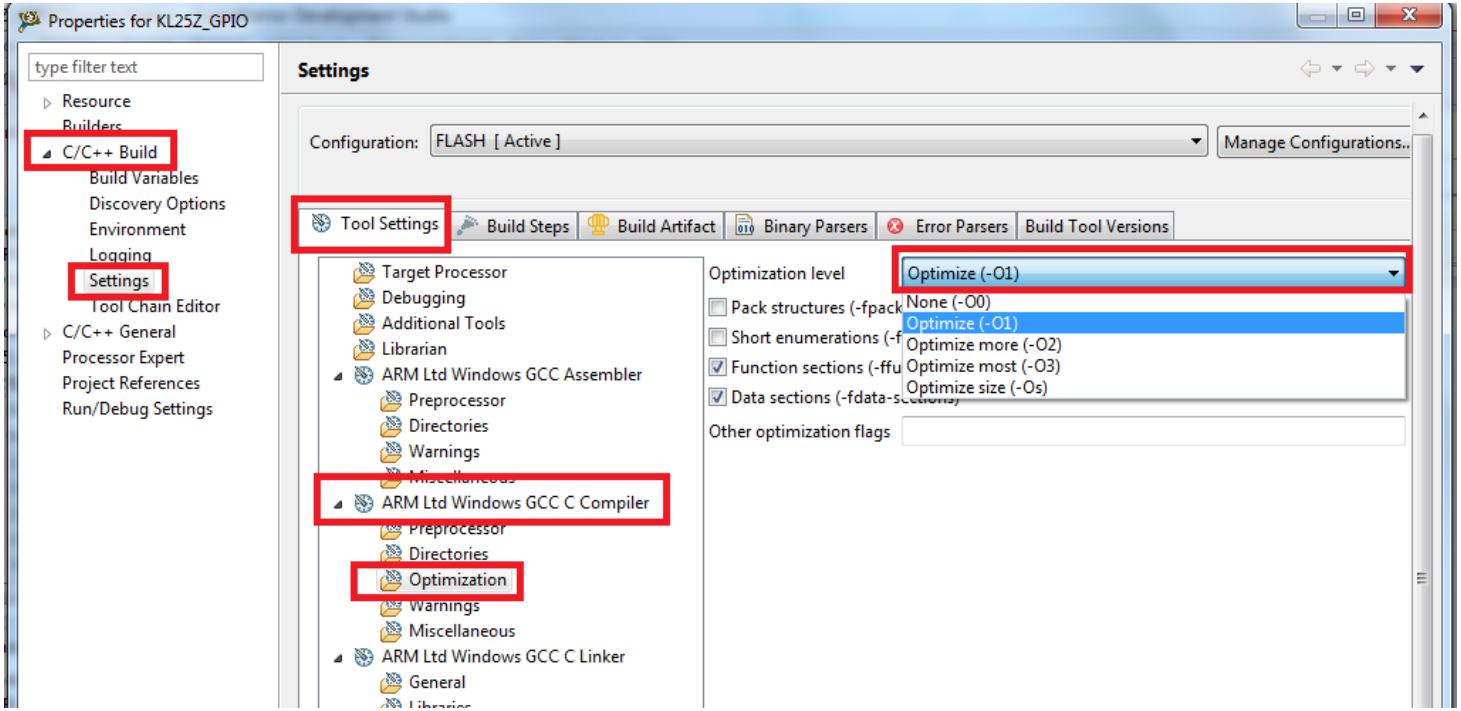
Single Cycle IO (Fast IO) - Result

- The GPIO state change from high to low that took 19.2n sec in one instruction.
- The core clock is 48MHz (20.8ns) , that mean only one instruction cycle required for once GPIO access .
- It's real zero wait state access .



Note

- With compiler optimizations set to “None(-O0)" each of the lines of C code may be compiled into several assembly statements to perform the write to the register, which takes multiple instruction cycles to perform .
- With compiler optimizations set to “Optimize(-O1)“ or higher , the register write only takes one assembly instruction.



SECURE CONNECTIONS
FOR A SMARTER WORLD