I2C

~~20140501~~
2015 12/16

Reference:
UM10204.pdf
i2c_utility_0.4.1.f

i2c-communication

i2c-communication need pull-up-resistors on scl/sda-lines.
Resistor-value is depended on total capasitance of i2c-devices on sda/scl-lines.
Normally, 10k-ohm is no problem.
For example,QuickStartboard and Demoboard,etc.

When there are many i2c-devices, total capacitance become bigger.
When capacitance on scl and sda-lines is too big, timing for i2c-signal become poor.
And propller can't load code inside eeprom even if executing 'reboot'.
So, Propeler reply nothing.

Prevending this phenominon, pull-up-resistor on scl and sda-lines should be small.
We need to add resistor on scl/sda-lines.
For example, combined resistance is 5k-ohm when adding 10k-ohm

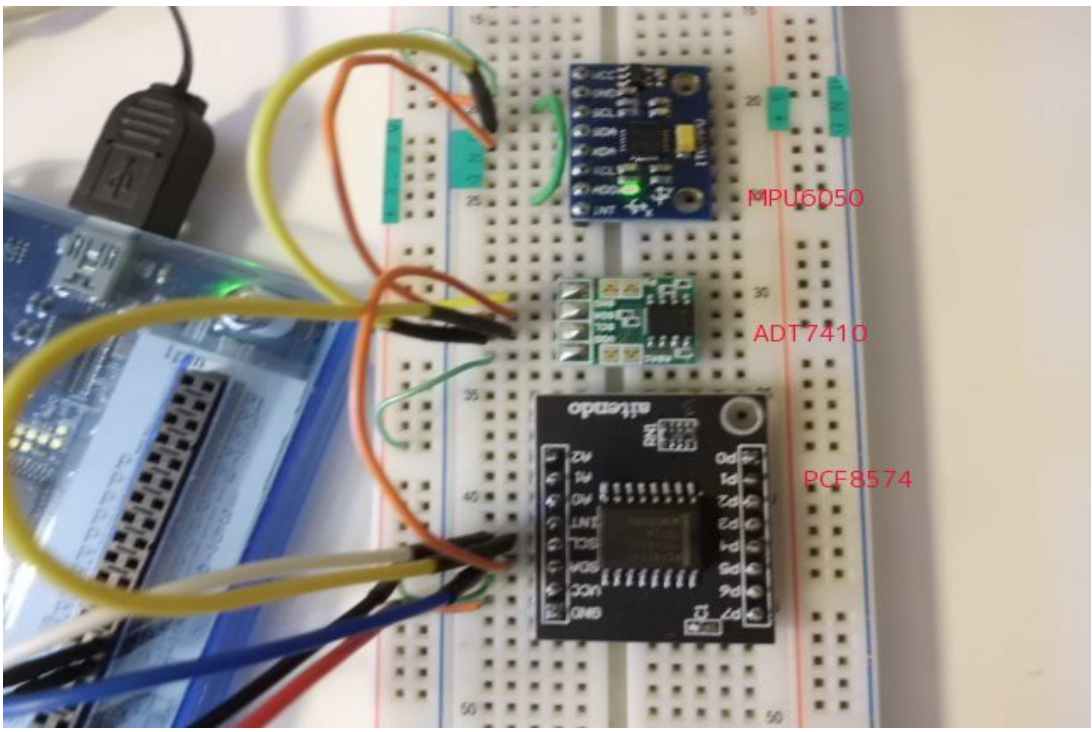
Setting sda(scl) to Hi; Set bit of sda(scl) to input
Setting sda(scl) to Lo; Set bit of sda(scl) to output [bit of outa for sda(scl) is '0']

Slave-address

Slave-address is 7bit.
(10bit-addres cannot use on i2c_utility_0.4.1.f)
When this address is sent to device, it is placed to [bit7-bit1].
In case of h20, it is b01000000(write:h40) at writing and it is b01000001(read:h41) at reading.
Please refer UM10204.pdf in case of 10bit-address.

About clock-stretch

i2c_utility_0.4.1.f don't treat clock-stretch.
Because I don't have i2c-device generate clock-stretch.
If I get i2c-device generate clock-stretch, then I re-write i2c-words.




```
Prop0 Cog6 ok
i2c_detect
   0 1 2 3 4 5 6 7 8 9 A B C D E F
00: 00 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- <-- ADT7410 reply to general-call--address
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: 20 -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- <-- PCF8574
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- 48 -- -- -- -- -- -- --  <-- ADT7410
50: 50 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --  <-- eeprom on QuickStartboard
60: -- -- -- -- -- -- -- -- 68 -- -- -- -- -- -- --  <-- MPU6050
70: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
i2c_device:4
[0 – 7] and [h78 – h7F] are reserve-address for 12c

Prop0 Cog6 ok
```

Bus clear

i2c-communication always is started by master.
But scl-line stuck Low in the unlikely event.
If i2c-device has HW reset input, reset by using it.
If nothing, do cycle power to device.

If sda-line stuck Low in the unlikely event,;
1 Sending nine clock pulse to device (word'bus_clr' inside i2c_utility_0.4.1.f)
2 Using HW reset input od device
3 Doing cycle power to device

About i2c_utility_0.4.1.f

_eestart

Firstly outa for sda is lo, and set output
Secondary outa for scl is lo, and set output

Sr

Repeated Start when using read-procedure
Set sda/scl to input (both line are Hi)
outa for sda is lo, and set output
outa for scl is lo, and set output

Sr is same as _eestart.
Device(MLX90614ESF-BAA) for SMBus caused Time-out error.
SMBus is almost same as I2C. But there is Time-out on SMBus.
Error occur because Sr is Forth-word.
_eestart(Sr) is re-defined as assembler-word.

_eestop

Stop i2c-commnication
Set scl to input
Set sda to input

i2c_wr

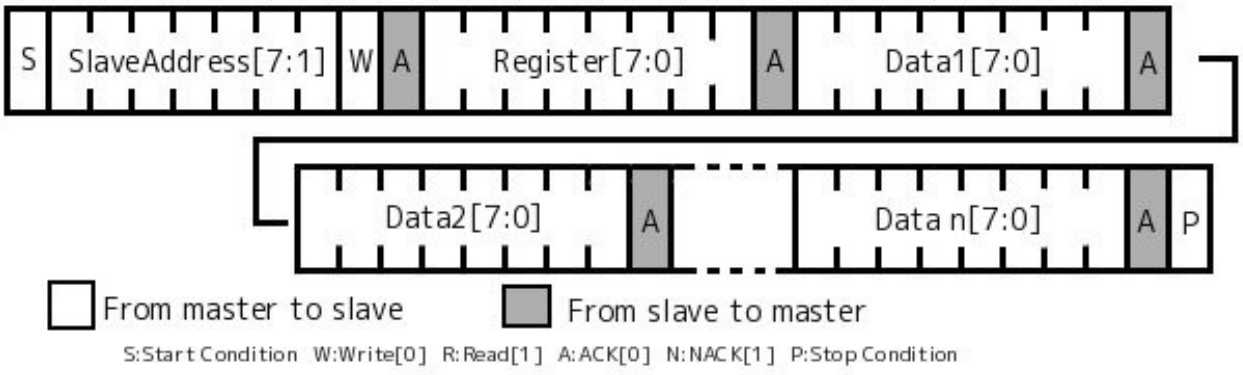
Writing 1byte to device's register

i2c_wr_multi

Writing multi-bytes to device's register

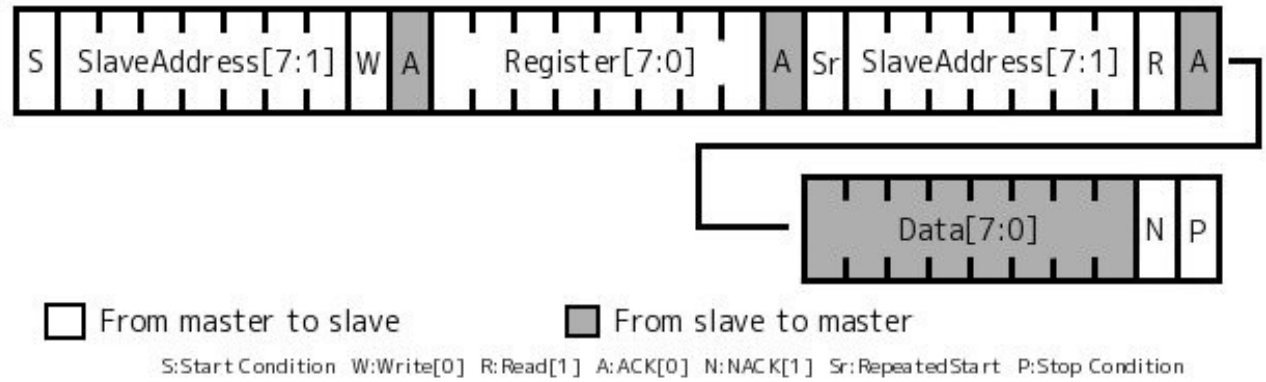
When this word is used, i2c_device need to have auto-increment for register.

Bytes is from 1 to n.

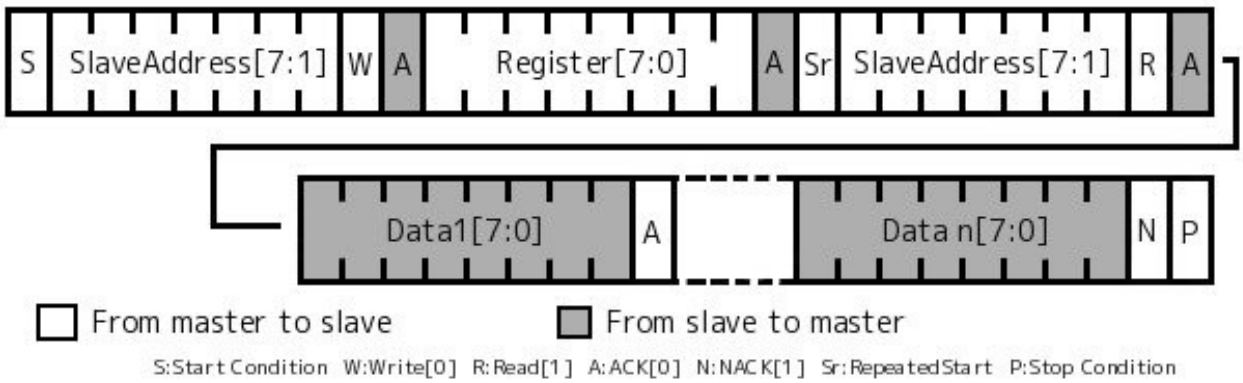


i2c_rd

Reading 1byte from device's register



i2c_rd_multi
Reading multi-bytes to device's register
Bytes is from 1 to n.

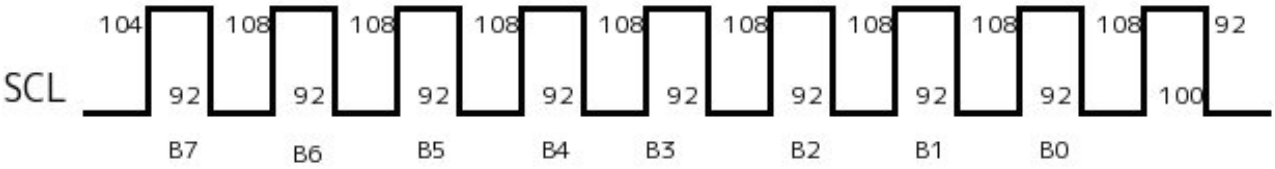


When this word is used, i2c_device need to have auto-increment for register.

Some i2c-devices don't have such a read/write-procedure.
So, some devices cannot use *i2c_wr*/*i2c_wr_multi*/*i2c_rd*/*i2c_rd_multi*.
And then, it can make procedure by using _eestart Sr _eestop _eewrite _eeread.

Clock for _eewrite is about 400kHz.

Number is ticks.

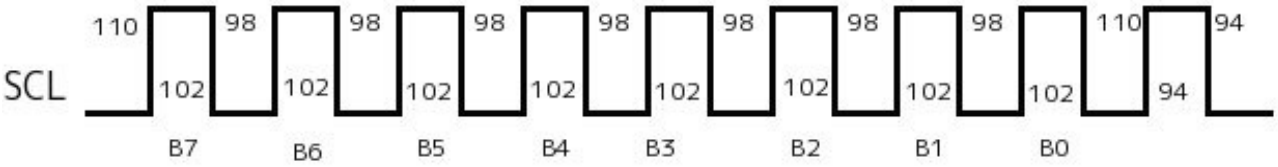



```
build_BootOpt :rasm
                    mov    $C_treg1 , # h8                4ticks    scl=Lo
\ --- Start tranmit 8bits
__1
                    \ Set data-bit to sda
                    test   $C_stTOS , # h80      wz        4ticks    scl=Lo
                    muxz   dira , __sda                    4ticks    scl=Lo
                    \ Wait 88ticks
                    jmpret  __delayret , # __delay         4ticks+84ticks    scl=Lo

                    \ Set scl to input(Hi)
                    andn   dira , __scl            4ticks    scl=Lo
                    \ Wait 88ticks
                    jmpret  __delayret , # __delay         4ticks+84ticks    scl=Hi

                    \ Set scl to output(lo)
                    or     dira , __scl                    4ticks    scl=Hi

                    \ Shift data-bit
                    shl    $C_stTOS , # 1                  4ticks    scl=Lo
                    \ Finish 8bit?
                    djnz   $C_treg1 , # __1       4ticks when jump  scl=Lo             8ticks when no-jump
\ --- Finish transmit 8bits

                    \ Set sda to input(floating)
                    andn   dira , __sda                    4ticks    scl=Lo
                    \ Wait 88ticks
                    jmpret  __delayret , # __delay         4ticks+84ticks    scl=Lo

                    \ Set scl to input(Hi)
                    andn   dira , __scl            4ticks    scl=Lo
                    \ Wait 88ticks
                    jmpret  __delayret , # __delay         4ticks+84ticks    scl=Hi
                    \ Input sda from slave,  Read ACK
                    test   __sda , ina   wz                4ticks    scl=Hi
                    muxnz  $C_stTOS , $C_fLongMask   4ticks       scl=Hi

                    \ Set scl to lo
                    or     dira , __scl                    4ticks    scl=Hi

                    \ Wait 88ticks
                    jmpret  __delayret , # __delay         4ticks+84ticks    scl=Lo

                    \ Set sda to lo and output
                    or     dira , __sda            4ticks    scl=Lo

                    jexit

\ this delay makes for a 84ticks on an 80 Mhz prop   [Xtal:5MHz]
__delay
                    mov     $C_treg2 , # d71               5+(84-18)=71
                    add     $C_treg2 , cnt
                    waitcnt   $C_treg2 , # 0
__delayret
                    ret
```

```
__sda
                    h20000000
__scl
                    h10000000

;asm _eewrite
```

Clock for _eeread is about 400kHz.

Number is ticks.




```
build_BootOpt :rasm
                    mov    $C_treg1 , $C_stTOS 4ticks    scl=Lo
                    mov    $C_stTOS , # 0                4ticks    scl=Lo
                    \ Set sda to input(floating)
                    andn   dira , __sda                  4ticks    scl=Lo
                    mov    $C_treg3 , # h8               4ticks    scl=Lo
\ --- Start tranmit 8bits
__1
                     \ Wait 90ticks
                    jmpret  __delayret , # __delay       4ticks+86ticks    scl=Lo
                    \ Set scl to input(Hi)
                    andn   dira , __scl           4ticks    scl=Lo
                    \ Wait 90ticks
                    jmpret  __delayret , # __delay       4ticks+86ticks    scl=Hi

                    test   __sda , ina   wc              4ticks    scl=Hi
                    rcl    $C_stTOS , # 1                4ticks    scl=Hi

                    \ Set scl to output(lo)
                    or     dira , __scl                  4ticks    scl=Hi

                    djnz   $C_treg3 , # __1      4ticks when jump  scl=Lo            8ticks when no jump scl=Lo
\ --- Finish transmit 8bits

                    cmp    $C_treg1 , # 0 wz             4ticks    scl=Lo

                    \ Set sda to hi if $C_treg is not 0
                    muxz   dira , __sda                  4ticks    scl=Lo
                    \ Wait 90ticks
                    jmpret  __delayret , # __delay       4ticks+86ticks    scl=Lo

                    \ Set scl to input(Hi)
                    andn   dira , __scl           4ticks    scl=Lo
                    \ Wait 90ticks
                    jmpret  __delayret , # __delay       4ticks+86ticks    scl=Hi

                    \ Set scl to output(lo)
                    or     dira , __scl                  4ticks    scl=Hi

                    \ Set sda to output(lo)
                    or     dira , __sda           4ticks    scl=Lo
                    \ Wait 90ticks
                    jmpret  __delayret , # __delay       4ticks+86ticks    scl=Lo

                    jexit

\ this delay makes for a 86ticks on an 80 Mhz prop  [Xtal:5MHz]
__delay
                    mov     $C_treg2 , # d73             5+(86-18)
                    add     $C_treg2 , cnt
                    waitcnt  $C_treg2 , # 0
__delayret
                    ret

__sda
                    h20000000
```

```
__scl
                    h10000000

;asm _eeread
```