Application Note

PRU Fast GPIO (RTOS)- AM243x LP EVM / AM64x GP EVM

Rev. <ref>
<date>

Revision History

Version	Date	Author	Description
0.2	Oct 18 – 2022		Added FAST GPIO example

Table of Contents

1 Fast_GPIO

Learning goal:

- Example explains requirement to setup pin-mux either from PRU, ARM or SYSCONFIG
- Example explains lock and unlock of pinmux
- Example explains padconfig settings and how they show up in R30 and R31, especially outputs which can be read back.
- Example show min latency of GPI copy to GPO and visualizes signal on scope

There are two types of GPIO on the Sitara device. System GPIOs can be programmed by any bus master on the SOC. The system GPIOs reside on peripheral interconnect which does arbitration for multiple masters to same end-point. Due to different frequency and bus width access to GPIO has certain latency. The second type of GPIOs reside in PRU register R30 and R31. There are 20 GPIs and GPOs pre PRU which share the same external pin. The pad configuration for certain functions can be configured during boot or run-time. The pad configuration registers are locked by default and a certain key need to be programmed to un-lock register access.

```
;******************** padconfig ************************
; unlock PADMMR config register
; partition 0
    ldi32    r2, 0x000f1008  ; LOCK0 KICK0 register
    ldi32    r3, 0x000f100c  ; LOCK0 KICK1 register
    ldi32    r4, 0x68EF3490  ; Kick 0
    ldi32    r5, 0xD172BC5A  ; kick 1

    sbbo     &r4, r2, 0, 4
    sbbo     &r5, r3, 0, 4

; partition 1
    ldi32    r2, 0x000f5008  ; LOCK1 KICK0 register
    ldi32    r3, 0x000f500c  ; LOCK1 KICK1 register

    sbbo     &r4, r2, 0, 4
    sbbo     &r5, r3, 0, 4

; pin-mux configuration - PRG0_PRU0_GPI1 - BP.32
; alternative system GPIO1_1
    ldi32    r2, 0x000F4164
 .if(PRU_GPI)
    ldi32    r3, 0x00040001
 .else
    ldi32    r3, 0x00040007
 .endif
    sbbo     &r3, r2, 0, 4
```

Figure 1 pad config from PRU code

The pad configuration code in figure 1 first unlocks RADMMR configuration registers. The same pin can be used as system GPIO1_1 or ICSS_G0_PRU0_GPI1. The pin as available on booster pack header BP.32. Last digit in r3 defines the mode of the pin. Mode 1 is used for PRU GPI and mode 7 for system GPIO.

```
;************** GPIO latency loop ************************
; follow GPI pin in PRU mode and ststem mode
; r31 bit 1 maps to PRU0_GPI1
; r30 bit 0 maps to PRU0_GPO0
;
; system GPIO offsets to base address 0x00601000
; 0x18 - set data
; 0x1C - clr data
; 0x20 - in_data

idle_loop:

; poll for rising edge
 .if(PRU_GPI)
   wbs      r31,  1
 .else
wait_high:
   lbbo     &r3,r2, 0x20 , 1
   qbbc     wait_high, r3.b0, 1
 .endif
; set GPO
 .if(PRU_GPO)
   set      r30, r30, 0
 .else
   sbbo     &r4.b0, r2, 0x18, 1
 .endif

 ; poll for falling edge
 .if(PRU_GPI)
   wbc      r31,  1
 .else
wait_low:
   lbbo     &r3,r2, 0x20 , 1
   qbbs     wait_low, r3.b0, 1
 .endif
; clear GPO
 .if(PRU_GPO)
   clr      r30, r30, 0
 .else
   sbbo     &r4.b0, r2, 0x1c, 1
 .endif
   qba      idle_loop
```

Figure 2 GPIO latency loop

The GPIO latency code in figure 2 has two options based on definition of PRU_GPI and PRU_GP0. The GPIO loop follows the signal on GPIO1_1 or PRU_GPI1. PRU can directly read status of that pin from register 31 bit 1. When defined as system GPIO then PRU needs to read from global memory address of GPIO peripheral. The difference in latency can be seen when comparing figure 3 with figure 4. PRU GPIOs are more than 10x faster than system GPIOs.

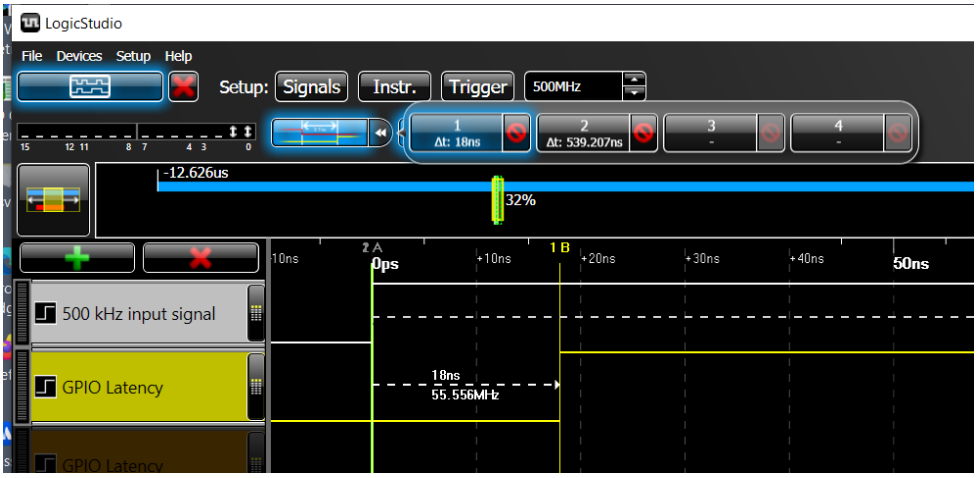

Figure 3 PRU GPIO latency

Figure 4 System GPIO latency