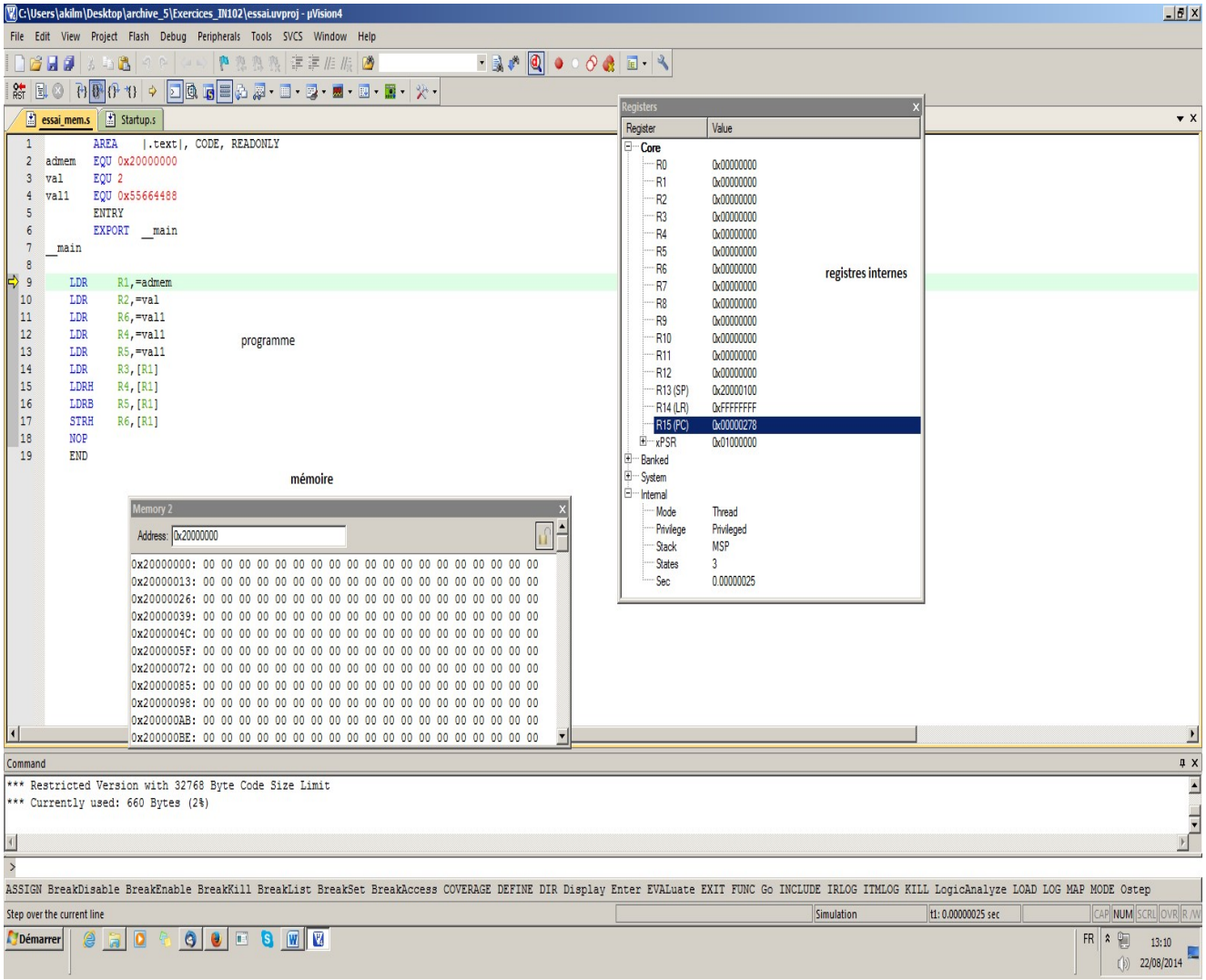
Kit EVALBOT et E/S

ROSTOM KACHOURI

ESIEE
PARIS

Kit EVALBOT et E/S

Contexte : Exécution d'un programme par un microprocesseur : exécution Instruction par Instruction, observation des registres internes, de la mémoire, des résultats obtenus (à différentes étapes d'un programme), tests et corrections. Ecriture de programmes, mise en œuvre de ces programmes en utilisant l'environnement de simulation KEIL - uVISION. Cette phase est importante pour programmer le ROBOT EVALBOT, afin de le « piloter » (faire avancer avec différents scénarios : suivre une trajectoire, détecter des obstacles, changer d'orientation, etc..).



Carte EVALBOT :
Gestion des ports d'Entrée –Sortie (IO)

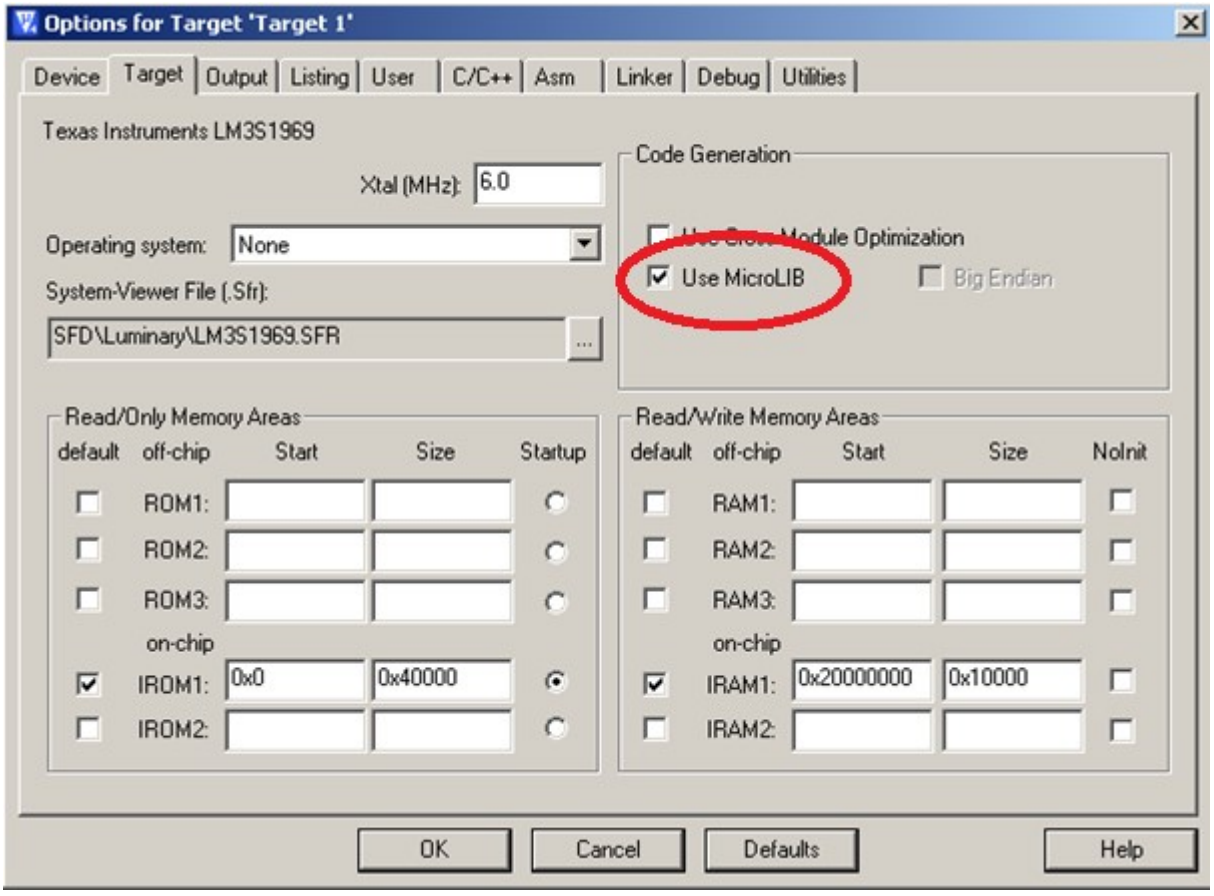
Les séances de travaux pratiques se déroulent en salle informatique – utilisation du simulateur *µVISION4* ou version supérieure sur PC et le kit EVALBOT. Il est nécessaire d'aller « au bout des choses » : identifier les erreurs, les corriger, programmer soi-même et s'auto former.

Comme pour les séances de TD avec le logiciel Keil sur PC, il faut créer un projet pour le processeur Texas Instrument :

- choisir la cible processeur « LM1968 » (l'Evalbot repose en réalité sur un « LM3S9B92 » mais le « LM1968 » est compatible et simulé)

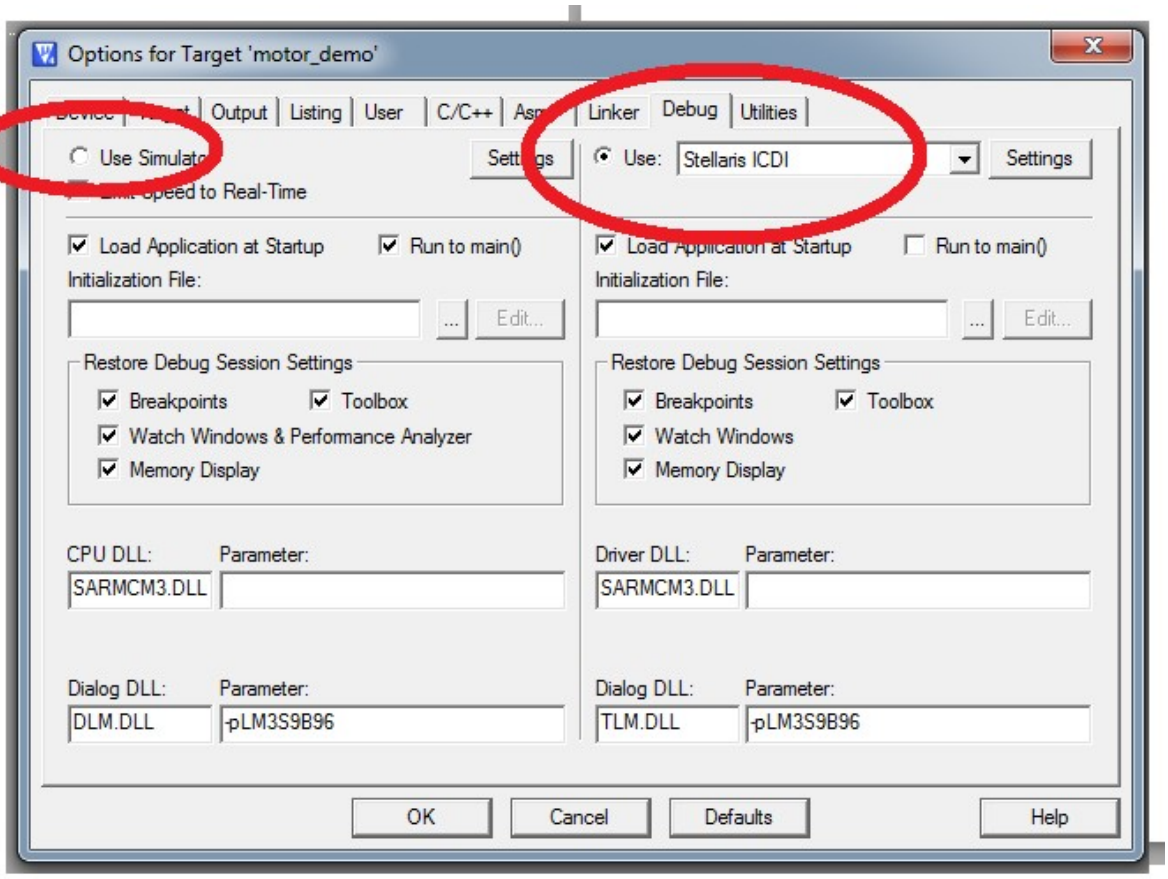
- autorisez la copie du fichier Startup.s proposé par défaut

- cocher « use microlib » dans l'onglet « Target » (pour ne pas rechercher les sections de code spécifique au C)

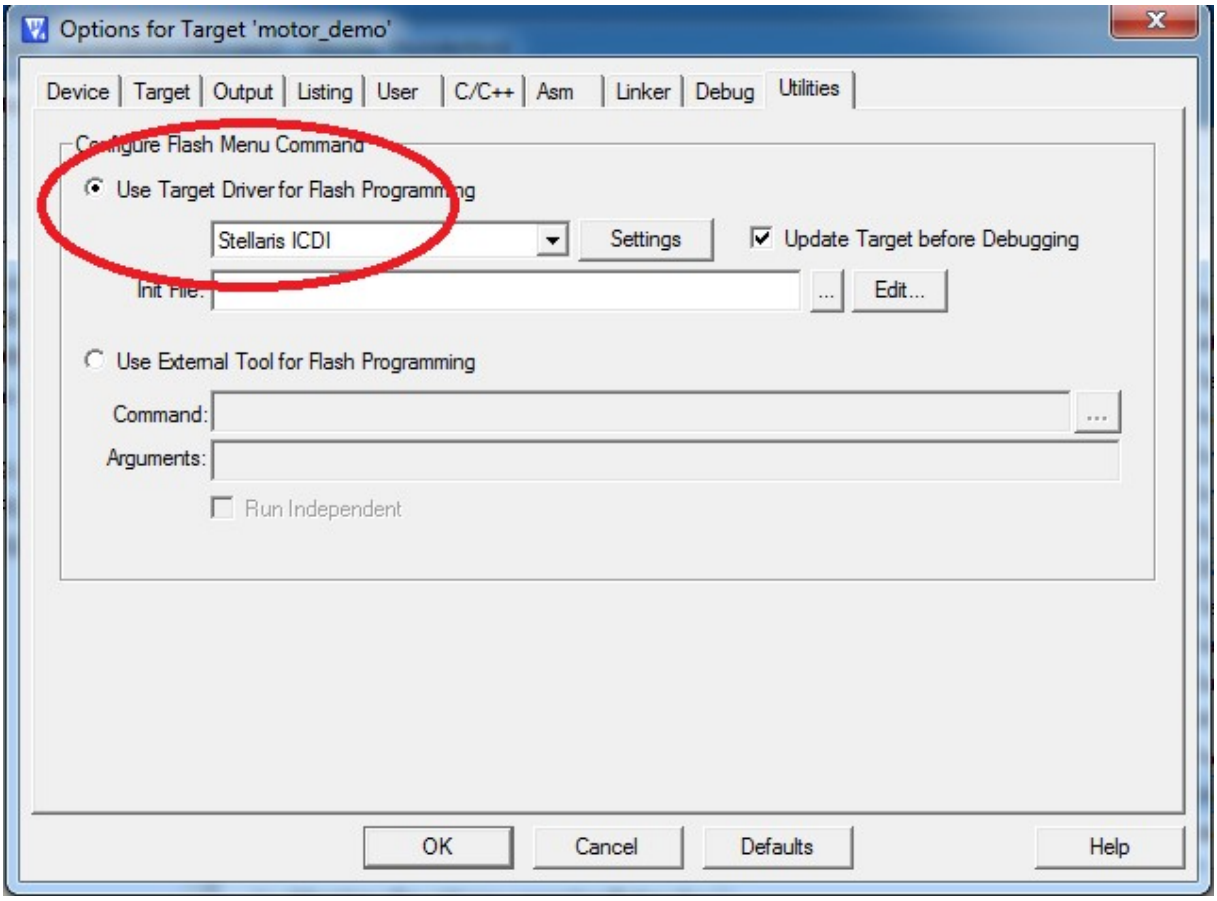


- puis dans les propriétés du projet il faut configurer ainsi pour utiliser l'interface USB de Texas :

  - dans onglet « Debug » : use Stellaris ICDI



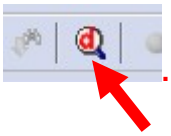
- dans l'onglet « Utilities » : Stellaris ICDI



Ensuite pour exécuter le code sur le kit Evalbot il faut :

1) Brancher l'Evalbot par le câble USB sur le bon port USB (voir page 4)
2) Compiler le code et vérifier qu'il n'y a pas d'erreur (Build All)
3) Allumer l'Evalbot : bouton ON, il faut alors attendre quelques secondes que le driver USB s'installe
4) Transférer le code vers la carte en appuyant sur « Load » :

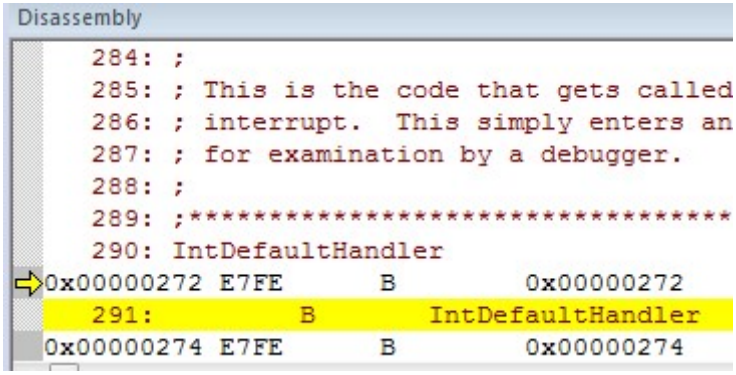
5) Si tout s'est bien passé vous obtenez le message suivant « Programming Done, Verify OK ». Le code est maintenant sur la carte il suffit d'appuyer à nouveau sur le bouton « ON » pour faire exécuter le programme qui vient d'être stocké dans l'Evalbot.

REMARQUE : quand le programme ne fonctionne pas vous pouvez travailler en mode pas à pas en cliquant sur « Debug » :  .

Cela transfère votre code dans la mémoire de l'Evalbot (il faut donc l'avoir allumé), puis vous pouvez avancer instruction par instruction en appuyant sur F11 et voir à chaque fois l'instruction qui va s'exécuter, l'état des registres, visualiser le contenu de la mémoire.

Vous pouvez aussi cliquez sur run puis sur stop  pour voir où en est votre programme.

Si quand vous faite stop vous constatez que le processeur est bloqué sur l'instruction suivante, cela veut dire qu'il y a eu un accès mémoire invalide.



– première partie - Prise en main :

Comprendre, exécuter et assimiler le programme permettant de faire clignoter La LED1, connectée à la broche 4 du port GPIOF. Voir code en annexe.

```
;; RK - Evalbot (Cortex M3 de Texas Instrument)
;; fait clignoter une seule LED connectée au port GPIOF

                AREA    |.text|, CODE, READONLY
; This register controls the clock gating logic in normal Run mode
SYSCTL_PERIPH_GPIOF  EQU  0x400FE108      ; SYSCTL_RCGC2_R (p291 datasheet
                                          ; de lm3s9b92.pdf)
; The GPIODATA register is the data register
GPIO_PORTF_BASE  EQU  0x40025000          ; GPIO Port F (APB) base:0x4002.5000
                                          ;(p416 datasheet de lm3s9B92.pdf)
; configure the corresponding pin to be an output
; all GPIO pins are inputs by default
GPIO_O_DIR   EQU 0x00000400  ; GPIO Direction (p417 datasheet de lm3s9B92.pdf)

; The GPIODR2R register is the 2-mA drive control register
; By default, all GPIO pins have 2-mA drive.
GPIO_O_DR2R      EQU  0x00000500          ; GPIO 2-mA Drive Select (p428
                                          ; datasheet de lm3s9B92.pdf)
; Digital enable register
; To use the pin as a digital input or output, the corresponding GPIODEN bit must be set.
GPIO_O_DEN       EQU  0x0000051C          ; GPIO Digital Enable (p437 datasheet de
                                          ; lm3s9B92.pdf)
; PIN select
PIN4             EQU      0x10            ; led1 sur broche 4

; blinking frequency
DUREE            EQU   0x002FFFFF         ; Random Value
```

```
        ENTRY
        EXPORT      __main
__main
; ;; Enable the Port F peripheral clock by setting bit 5 (0x20 == 0b1 0  0  0  0 0)
; ;;                                                    (GPIO::F E D C B A)
;;; (p291 datasheet de lm3s9B96.pdf)
        ldr r6, = SYSCTL_PERIPH_GPIOF           ;; RCGC2
        mov r0, #0x00000020                     ;; Enable clock sur GPIO F où sont
                                        ; ;; branchés les leds (0x20 == 0b1 0  0  0  0  0)
                                        ; ;;                (GPIO::F E D C B A)
        str r0, [r6]

; ;; "There must be a delay of 3 system clocks before any GPIO reg. access  (p413 datasheet
; ;;  de lm3s9B92.pdf)
        nop                                             ;; très très important....
        nop
        nop                             ;; pas necessaire en simu ou en debbug step by step...

;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^CONFIGURATION LED
        ldr r6, = GPIO_PORTF_BASE+GPIO_O_DIR    ;; 1 Pin du portF en sortie
        ldr r0, = PIN4
         str r0, [r6]

        ldr r6, = GPIO_PORTF_BASE+GPIO_O_DEN     ;; Enable Digital Function
        ldr r0, = PIN4
        str r0, [r6]
```

```
        ldr r6, = GPIO_PORTF_BASE+GPIO_O_DR2R  ;; Intensité de sortie (2mA)
        ldr r0, = PIN4
        str r0, [r6]

        mov r2, #0x000                  ;; pour éteindre LED
; allumer la led broche 4 (PIN4)
        mov r3, #PIN4                   ;; portF broche 4 : 00010000

;; @data Register = @base + (mask<<2) ==> LED1
        ldr r6, = GPIO_PORTF_BASE + (PIN4<<2)

;vvvvvvvvvvvvvvvvvvvvvvvvFin configuration LED

;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^CLIGNOTTEMENT
loop
        str r2, [r6]            ;; Eteint LED car r2 = 0x00
        ldr r1, = DUREE         ;; pour la durée de la boucle d'attente1 (wait1)
wait1   subs r1, #1
        bne wait1
        str r3, [r6]            ;; Allume portF broche 4 : 00010000 (contenu de r3)
        ldr r1, = DUREE         ;; pour la duree de la boucle d'attente2 (wait2)
wait2   subs r1, #1
        bne wait2
        b loop
        nop
        END
```

– deuxième partie – Exercices :

Il s'agit à la fin de cette séance d'être capable :

- D'écrire le programme correspondant à chaque exercice ci-dessous
- D'exécuter chaque programme et de montrer que le résultat rendu correspond à celui attendu
- D'expliquer le fonctionnement de chaque programme.

Exercices à faire : Écrire en langage d'assemblage ARM Cortex – M3 les programmes suivants. Exécuter ces programmes et vérifier leur bon fonctionnement sur la carte « Stellaris EVALBOT ».

1. Programme permettant de faire clignoter les deux LED1 et 2, connectées respectivement aux broches 4 et 5 du port GPIOF.

2. Programme permettant de faire allumer les deux LED1 et 2, un appui sur l'un des 2 pare-chocs (bumper) de devant éteint la LED correspondante.

3. Même question que le 2. Les pare-chocs (bumper) sont remplacés par les deux boutons poussoirs SW1 et SW2.

4. Programme permettant de garder les deux LED1 et LED2 éteintes, un appui sur l'un des 2 boutons poussoirs allume la LED correspondante.

5. Programme permettant de commander une LED à l'aide d'un bouton poussoir, un appuie sur le bouton poussoir SW1 allume la LED1, un deuxième appui sur le même bouton l'éteint.

6. Même question que le 5 en commandant les deux LEDs. Un appui sur le bouton poussoir SW1 allume les deux LEDs, un deuxième appui les éteint.

7. Programme permettant de changer inversement l'état des deux LEDs. Un appui sur le bouton poussoir SW1 allume la LED1 et éteint la LED2, un deuxième appui change l'état des deux LEDs.

Register 1: GPIO Data (GPIODATA), offset 0x000

The **GPIODATA** register is the data register. In software control mode, values written in the **GPIODATA** register are transferred onto the GPIO port pins if the respective pins have been configured as outputs through the **GPIO Direction (GPIODIR)** register (see page 418).

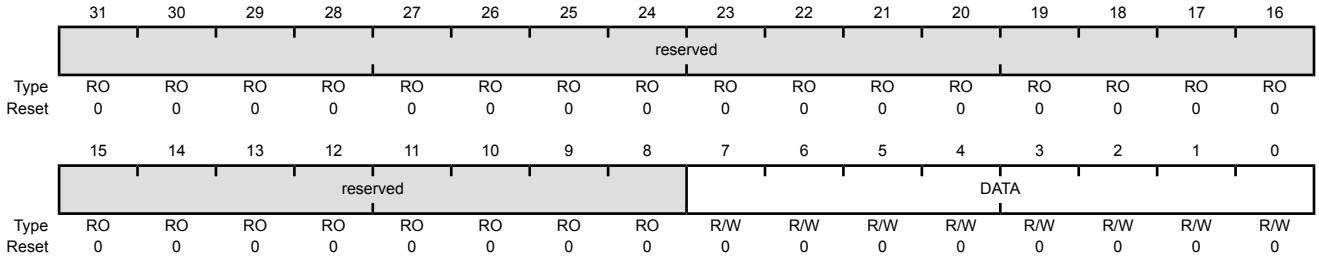
In order to write to **GPIODATA**, the corresponding bits in the mask, resulting from the address bus bits [9:2], must be set. Otherwise, the bit values remain unchanged by the write.

Similarly, the values read from this register are determined for each bit by the mask bit derived from the address used to access the data register, bits [9:2]. Bits that are set in the address mask cause the corresponding bits in **GPIODATA** to be read, and bits that are clear in the address mask cause the corresponding bits in **GPIODATA** to be read as 0, regardless of their value.

A read from **GPIODATA** returns the last bit value written if the respective pins are configured as outputs, or it returns the value on the corresponding input pin when these are configured as inputs. All bits are cleared by a reset.

GPIO Data (GPIODATA)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (APB) base: 0x4002.7000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (APB) base: 0x4003.D000
GPIO Port J (AHB) base: 0x4006.0000
Offset 0x000
Type R/W, reset 0x0000.0000



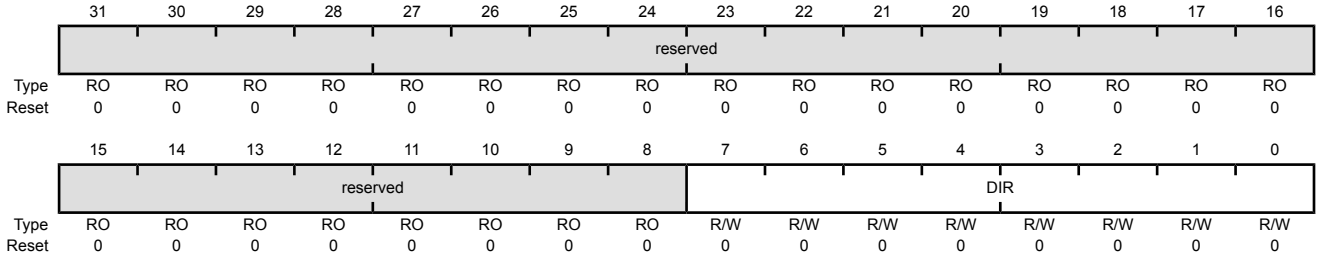
Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DATA	R/W	0x00	GPIO Data This register is virtually mapped to 256 locations in the address space. To facilitate the reading and writing of data to these registers by independent drivers, the data read from and written to the registers are masked by the eight address lines [9:2]. Reads from this register return its current state. Writes to this register only affect bits that are not masked by ADDR[9:2] and are configured as outputs. See “Data Register Operation” on page 410 for examples of reads and writes.

Register 2: GPIO Direction (GPIODIR), offset 0x400

The **GPIODIR** register is the data direction register. Setting a bit in the **GPIODIR** register configures the corresponding pin to be an output, while clearing a bit configures the corresponding pin to be an input. All bits are cleared by a reset, meaning all GPIO pins are inputs by default.

GPIO Direction (GPIODIR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (APB) base: 0x4002.7000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (APB) base: 0x4003.D000
GPIO Port J (AHB) base: 0x4006.0000
Offset 0x400
Type R/W, reset 0x0000.0000



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DIR	R/W	0x00	GPIO Data Direction

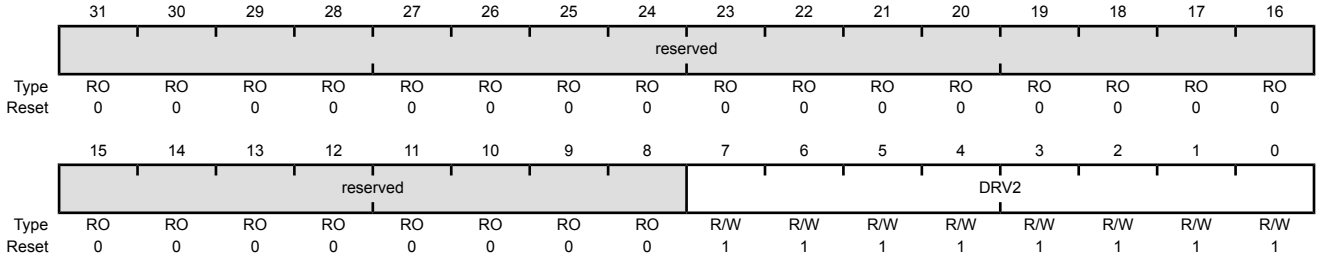
Value	Description
0	Corresponding pin is an input.
1	Corresponding pins is an output.

Register 11: GPIO 2-mA Drive Select (GPIODR2R), offset 0x500

The **GPIODR2R** register is the 2-mA drive control register. Each GPIO signal in the port can be individually configured without affecting the other pads. When setting the DRV2 bit for a GPIO signal, the corresponding DRV4 bit in the **GPIODR4R** register and DRV8 bit in the **GPIODR8R** register are automatically cleared by hardware. By default, all GPIO pins have 2-mA drive.

GPIO 2-mA Drive Select (GPIODR2R)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (APB) base: 0x4002.7000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (APB) base: 0x4003.D000
GPIO Port J (AHB) base: 0x4006.0000
Offset 0x500
Type R/W, reset 0x0000.00FF



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DRV2	R/W	0xFF	Output Pad 2-mA Drive Enable

Value	Description
1	The corresponding GPIO pin has 2-mA drive.
0	The drive for the corresponding GPIO pin is controlled by the **GPIODR4R** or **GPIODR8R** register.

Setting a bit in either the **GPIODR4** register or the **GPIODR8** register clears the corresponding 2-mA enable bit. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle.

Register 18: GPIO Digital Enable (GPIODEN), offset 0x51C

Note: Pins configured as digital inputs are Schmitt-triggered.

The **GPIODEN** register is the digital enable register. By default, all GPIO signals except those listed below are configured out of reset to be undriven (tristate). Their digital function is disabled; they do not drive a logic value on the pin and they do not allow the pin voltage into the GPIO receiver. To use the pin as a digital input or output (either GPIO or alternate function), the corresponding GPIODEN bit must be set.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

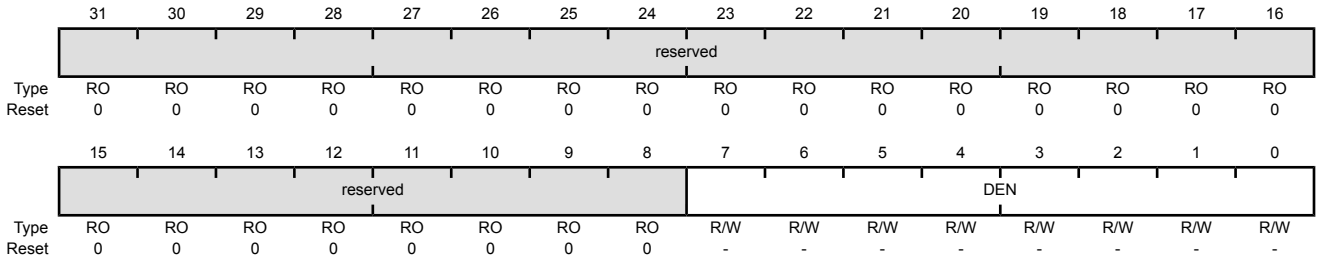
Table 8-11. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Note: The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the NMI pin (PB7) and the four JTAG/SWD pins (PC[3:0]). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 427), **GPIO Pull Up Select (GPIOPUR)** register (see page 433), **GPIO Pull-Down Select (GPIOPDR)** register (see page 435), and **GPIO Digital Enable (GPIODEN)** register (see page 438) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 440) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 441) have been set.

GPIO Digital Enable (GPIODEN)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (APB) base: 0x4002.7000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (APB) base: 0x4003.D000
GPIO Port J (AHB) base: 0x4006.0000
Offset 0x51C
Type R/W, reset -



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	DEN	R/W	-	Digital Enable

Value	Description
0	The digital functions for the corresponding pin are disabled.
1	The digital functions for the corresponding pin are enabled.

The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 8-1 on page 404.

Register 15: GPIO Pull-Up Select (GPIOPUR), offset 0x510

The **GPIOPUR** register is the pull-up control register. When a bit is set, a weak pull-up resistor on the corresponding GPIO signal is enabled. Setting a bit in **GPIOPUR** automatically clears the corresponding bit in the **GPIO Pull-Down Select (GPIOPDR)** register (see page 434). Write access to this register is protected with the **GPIOCR** register. Bits in **GPIOCR** that are cleared prevent writes to the equivalent bit in this register.

Important: All GPIO pins are configured as GPIOs and tri-stated by default (**GPIOAFSEL**=0, **GPIODEN**=0, **GPIOPDR**=0, **GPIOPUR**=0, and **GPIOPCTL**=0, with the exception of the pins shown in the table below. A Power-On-Reset ($\overline{\text{POR}}$) or asserting $\overline{\text{RST}}$ puts the pins back to their default state.

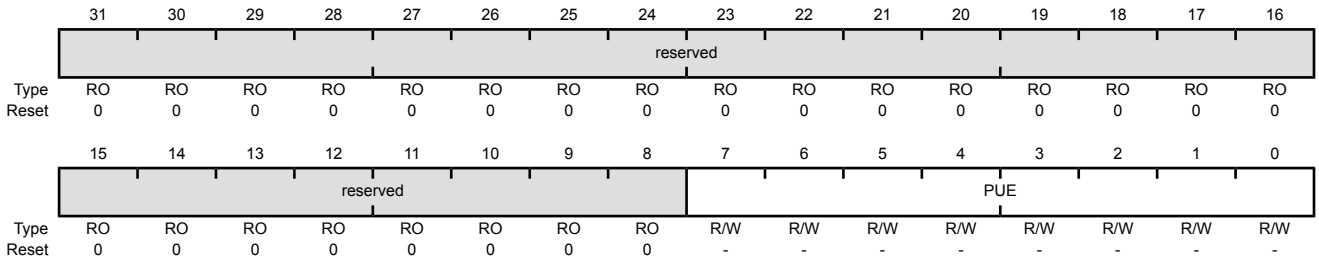
Table 8-9. GPIO Pins With Non-Zero Reset Values

GPIO Pins	Default State	GPIOAFSEL	GPIODEN	GPIOPDR	GPIOPUR	GPIOPCTL
PA[1:0]	UART0	0	0	0	0	0x1
PA[5:2]	SSI0	0	0	0	0	0x2
PB[3:2]	I^2C0	0	0	0	0	0x3
PC[3:0]	JTAG/SWD	1	1	0	1	0x1

Note: The GPIO commit control registers provide a layer of protection against accidental programming of critical hardware peripherals. Protection is provided for the `NMI` pin (`PB7`) and the four JTAG/SWD pins (`PC[3:0]`). Writes to protected bits of the **GPIO Alternate Function Select (GPIOAFSEL)** register (see page 426), **GPIO Pull Up Select (GPIOPUR)** register (see page 432), **GPIO Pull-Down Select (GPIOPDR)** register (see page 434), and **GPIO Digital Enable (GPIODEN)** register (see page 437) are not committed to storage unless the **GPIO Lock (GPIOLOCK)** register (see page 439) has been unlocked and the appropriate bits of the **GPIO Commit (GPIOCR)** register (see page 440) have been set.

GPIO Pull-Up Select (GPIOPUR)

GPIO Port A (APB) base: 0x4000.4000
GPIO Port A (AHB) base: 0x4005.8000
GPIO Port B (APB) base: 0x4000.5000
GPIO Port B (AHB) base: 0x4005.9000
GPIO Port C (APB) base: 0x4000.6000
GPIO Port C (AHB) base: 0x4005.A000
GPIO Port D (APB) base: 0x4000.7000
GPIO Port D (AHB) base: 0x4005.B000
GPIO Port E (APB) base: 0x4002.4000
GPIO Port E (AHB) base: 0x4005.C000
GPIO Port F (APB) base: 0x4002.5000
GPIO Port F (AHB) base: 0x4005.D000
GPIO Port G (APB) base: 0x4002.6000
GPIO Port G (AHB) base: 0x4005.E000
GPIO Port H (APB) base: 0x4002.7000
GPIO Port H (AHB) base: 0x4005.F000
GPIO Port J (APB) base: 0x4003.D000
GPIO Port J (AHB) base: 0x4006.0000
Offset 0x510
Type R/W, reset -



Bit/Field	Name	Type	Reset	Description
31:8	reserved	RO	0x0000.00	Software should not rely on the value of a reserved bit. To provide compatibility with future products, the value of a reserved bit should be preserved across a read-modify-write operation.
7:0	PUE	R/W	-	Pad Weak Pull-Up Enable Value Description 1 The corresponding pin has a weak pull-up resistor. 0 The corresponding pin is not affected. Setting a bit in the **GPIOPDR** register clears the corresponding bit in the **GPIOPUR** register. The change is effective on the second clock cycle after the write if accessing GPIO via the APB memory aperture. If using AHB access, the change is effective on the next clock cycle. The reset value for this register is 0x0000.0000 for GPIO ports that are not listed in Table 8-1 on page 403.