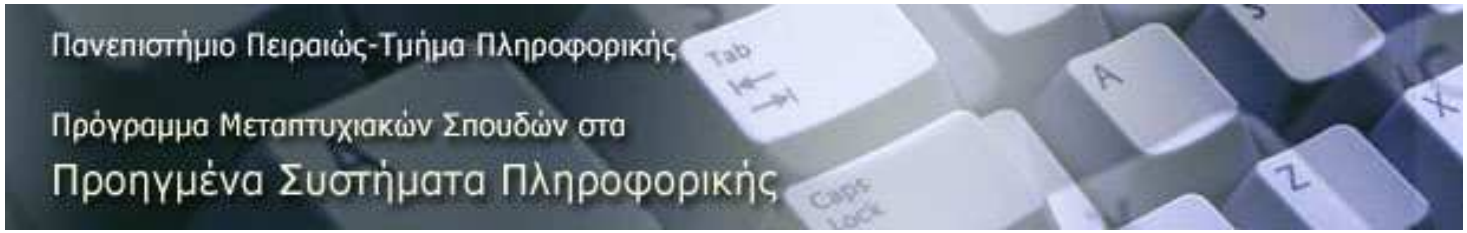
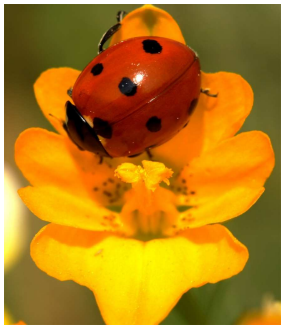


Ενσωματωμένα Υπολογιστικά Συστήματα

Μικροελεγκτές Atmel AVR32



Δάκης Παυλίδης, Εργαστηριακό Διδακτικό Προσωπικό
Μιχάλης Ψαράκης, Επίκουρος Καθηγητής

Δεκέμβριος 2013

Οικογένειες και χαρακτηριστικά Μικροελεγκτών AVR32

AT32AP7xxx (Ταχύτητα,MMU)

AT32UC3Bxxxx (Με Shadow Registers)

AT32UC3Axxxx (Χωρίς Shadow Registers)

AT32UC3Lxxx (Χαμηλή Κατανάλωση)

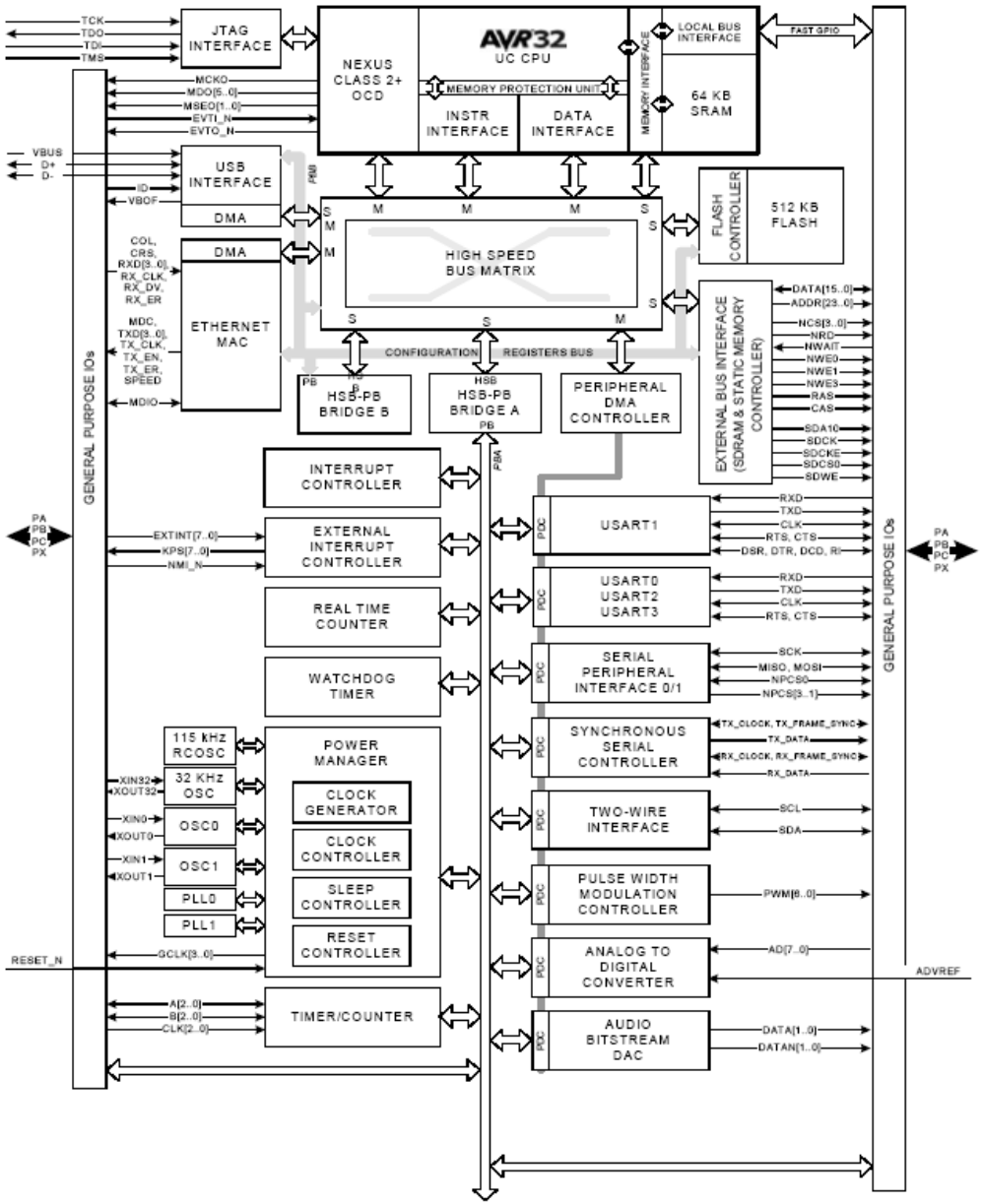
Βασική αρχιτεκτονική

Αρχιτεκτονική RISC Harvard 32 bits
Χρονισμός 33MHz (0-wait state)
66MHz (1-wait state)
Μνήμη Προγράμματος FLASH
Μνήμη Δεδομένων SRAM
Επέκταση με εξωτερική Μνήμη
Χειρισμοί Διακοπών 4 επιπέδων

Περιφερειακά

Power Manager
Watchdog Timer
Real Time Clock
Timer/Counter (x3)
PWM (x7)
ADC 10bits (x8)
DAC 16bit (x2)
Ethernet
USB
USART (x4)
SPI (x2)
SSI
TWI
JTAG

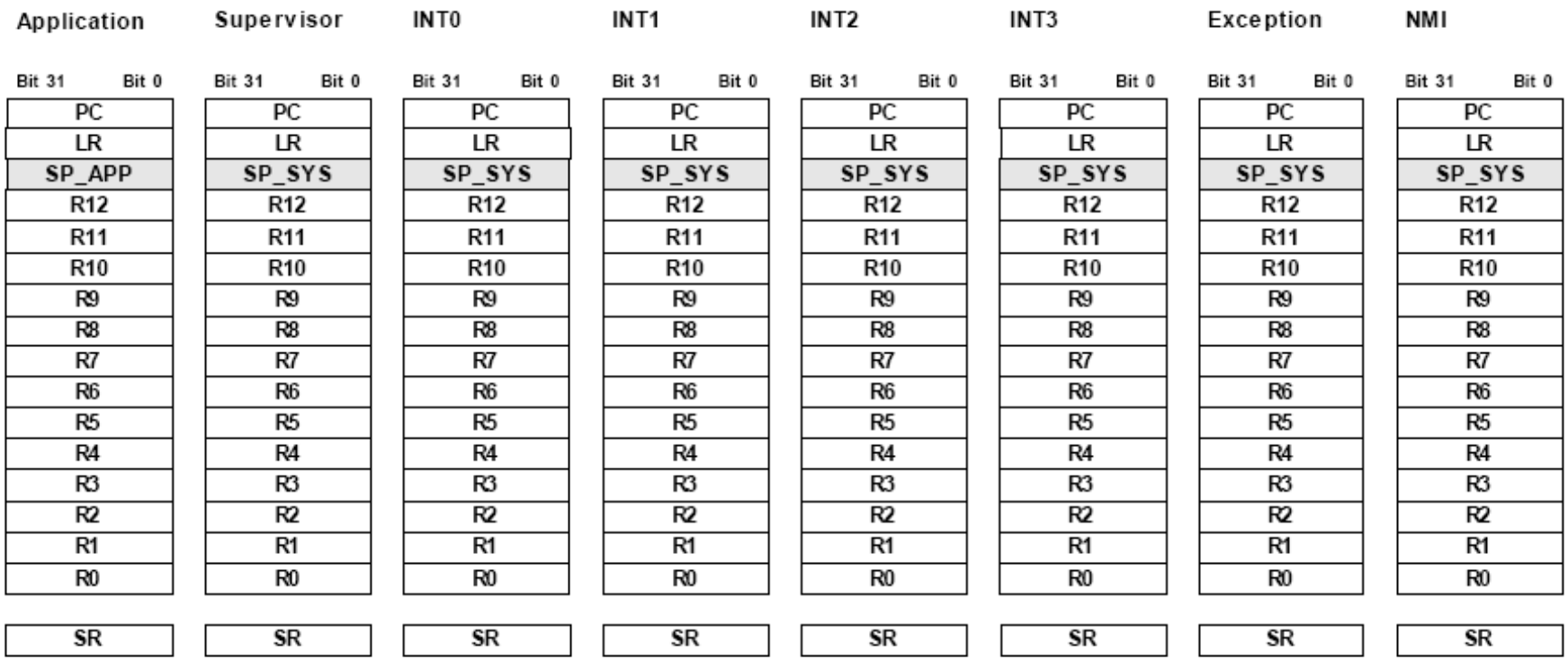
Γενική άποψη AT32UC3A



Σειρά AT32UC3Axxxx

Εξάρτημα	Μνήμη Προγράμματος (Kbytes)	Μνήμη Δεδομένων (Kbytes)	Εξωτερική Μνήμη	Συσκευασία Ακροδέκτες
AT32UC3A0128	128	32	NAI	LQFP 144
AT32UC3A0256	256	64	NAI	LQFP 144
AT32UC3A0512	**512**	**64**	**NAI**	**LQFP 144**
AT32UC3A1128	128	32	OXI	TQFP 100
AT32UC3A1256	256	64	OXI	TQFP 100
AT32UC3A1512	512	64	OXI	TQFP 100
AT32UC3A364	64	128	NAI	LQFP 144
AT32UC3A3128	128	128	NAI	LQFP 144
AT32UC3A3256	256	128	NAI	LQFP 144

Καταχωρητές Γενικού Σκοπού
Καταστάσεις Λειτουργίας



Γενικός Χάρτης Μνήμης

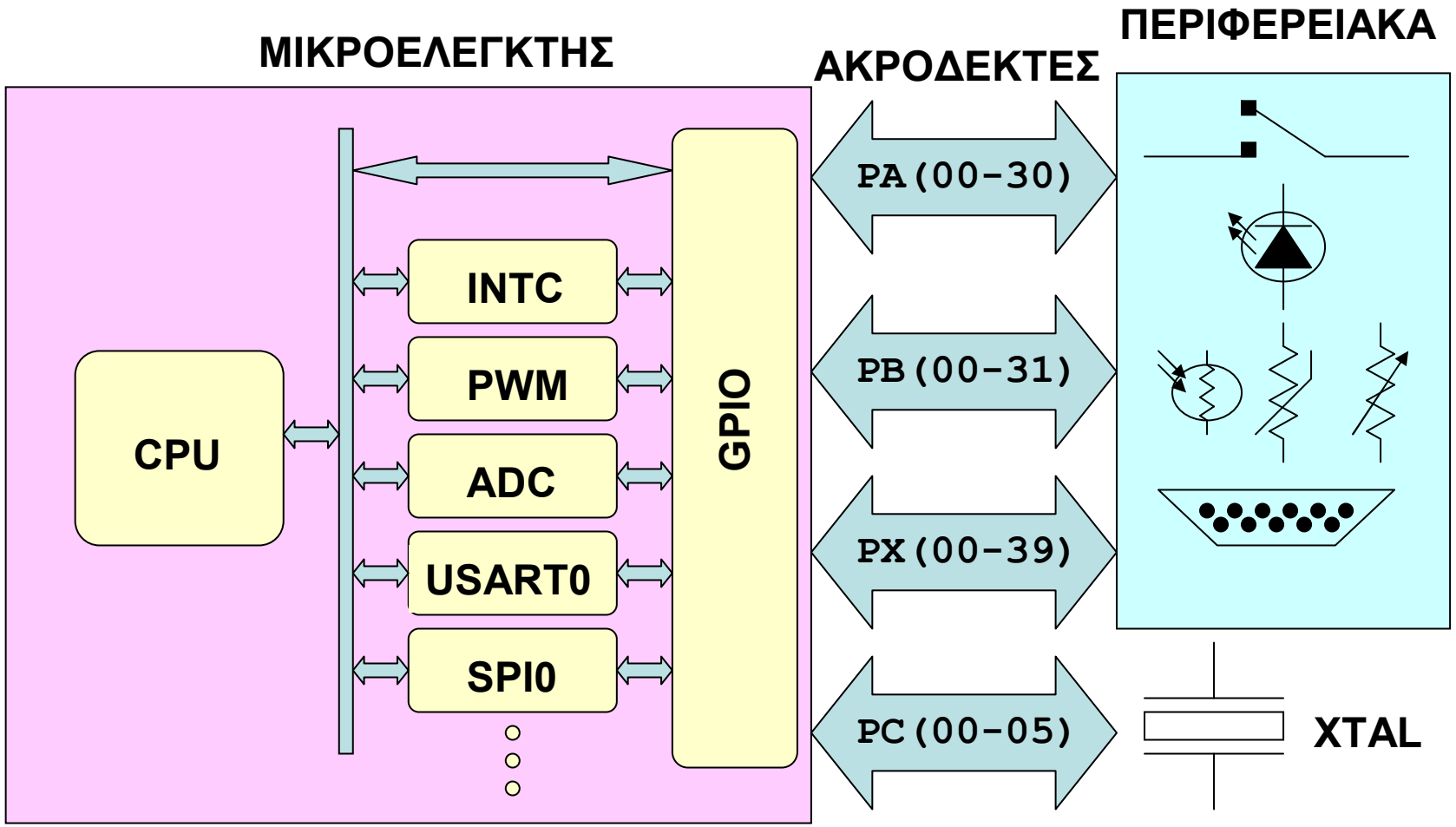
Περιφερειακή Συσκευή	Διεύθυνση Βάσης	Μέγεθος
Embedded SRAM	0x00000000	64 kbytes
Embedded Flash	**0x80000000**	512 kbytes
SRAM CS0	0xC0000000	16 Mbytes
SRAM CS2	0xC8000000	16 Mbytes
SRAM CS3	0xCC000000	16 Mbytes
SRAM CS1 / SDRAM CS0	0xD0000000	128 Mbytes
USB Configuration	0xE0000000	64 kbytes
Peripheral Bus A	**0xFFFF0000 ?**	64 kbytes
Peripheral Bus B	**0xFFFE0000 ?**	64 kbytes

Χάρτης Μνήμης Περιφερειακών

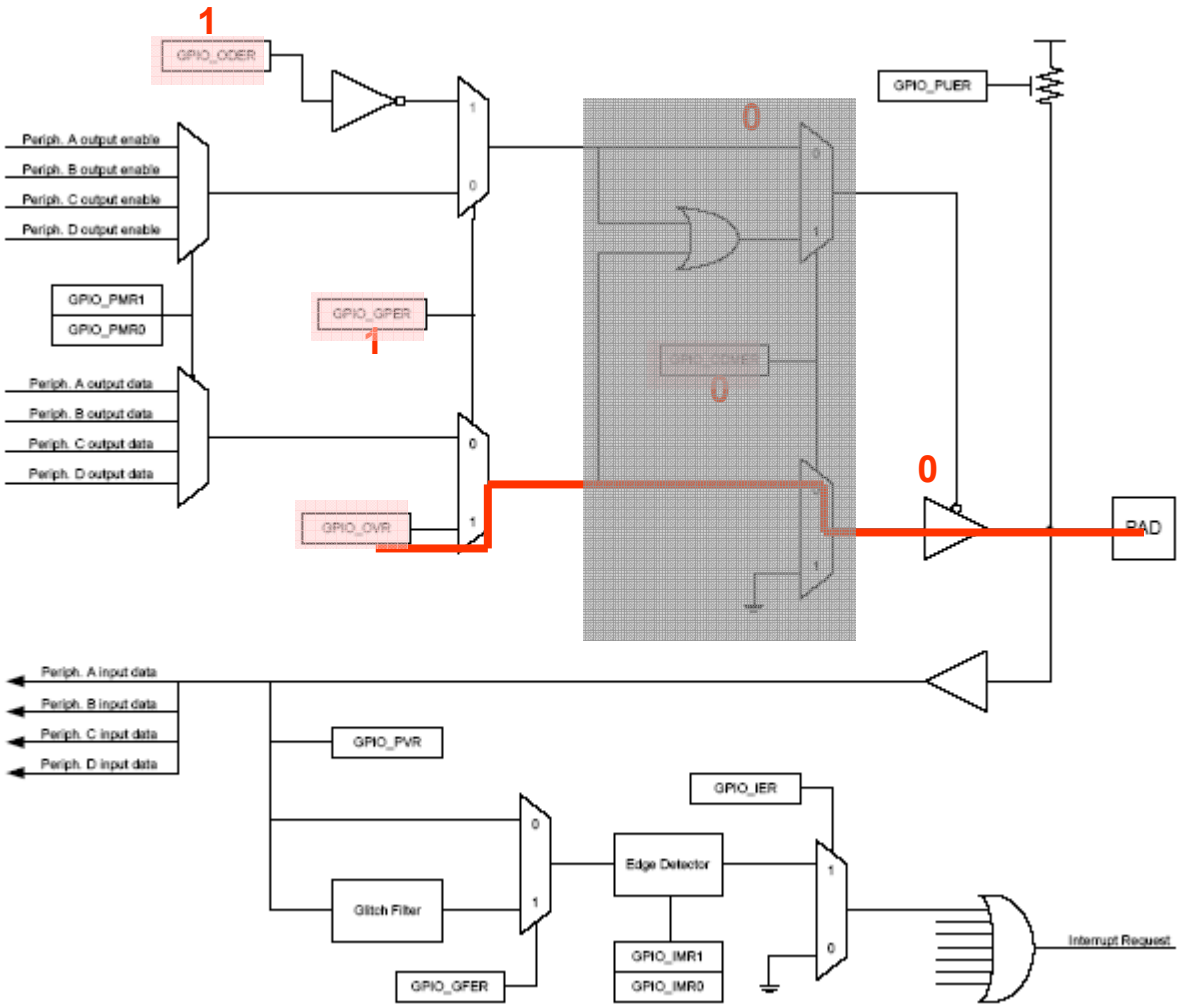
Περιφερειακή Συσκευή	Διεύθυνση Βάσης	Δίαυλος
USB Slave	0xE0000000	HSB
USB Configuration	0xFFFE0000	PBB
High Speed Matrix	0xFFFE1000	PBB
Flash Controller	0xFFFE1400	PBB
MAC Configuration	0xFFFE1800	PBB
SRAM Controller	0xFFFE1C00	PBB
SDRAM Controller	0xFFFE2000	PBB
Peripheral DMA Controller	0xFFFF0000	PBA
Interrupt Controller	0xFFFF0800	PBA
Power Manager	0xFFFF0C00	PBA
Real Time Clock	0xFFFF0D00	PBA
WatchDog Timer	0xFFFF0D30	PBA

Περιφερειακή Συσκευή	Διεύθυνση Βάσης	Δίαυλος
External Interrupt Controller	0xFFFF0D80	PBA
General Purpose IO Controller	0xFFFF1000	PBA
USART0	0xFFFF1400	PBA
USART1	0xFFFF1800	PBA
USART2	0xFFFF1C00	PBA
USART3	0xFFFF2000	PBA
Serial Peripheral Interface 0	0xFFFF2400	PBA
Serial Peripheral Interface 1	0xFFFF2800	PBA
Two Wire Interface	0xFFFF2C00	PBA
Pulse Width Modulation	0xFFFF3000	PBA
Synchronous Serial Controller	0xFFFF3400	PBA
Timer Counter	0xFFFF3800	PBA
Analog to Digital Converter	0xFFFF3C00	PBA

Διασύνδεση με εξωτερικά κυκλώματα

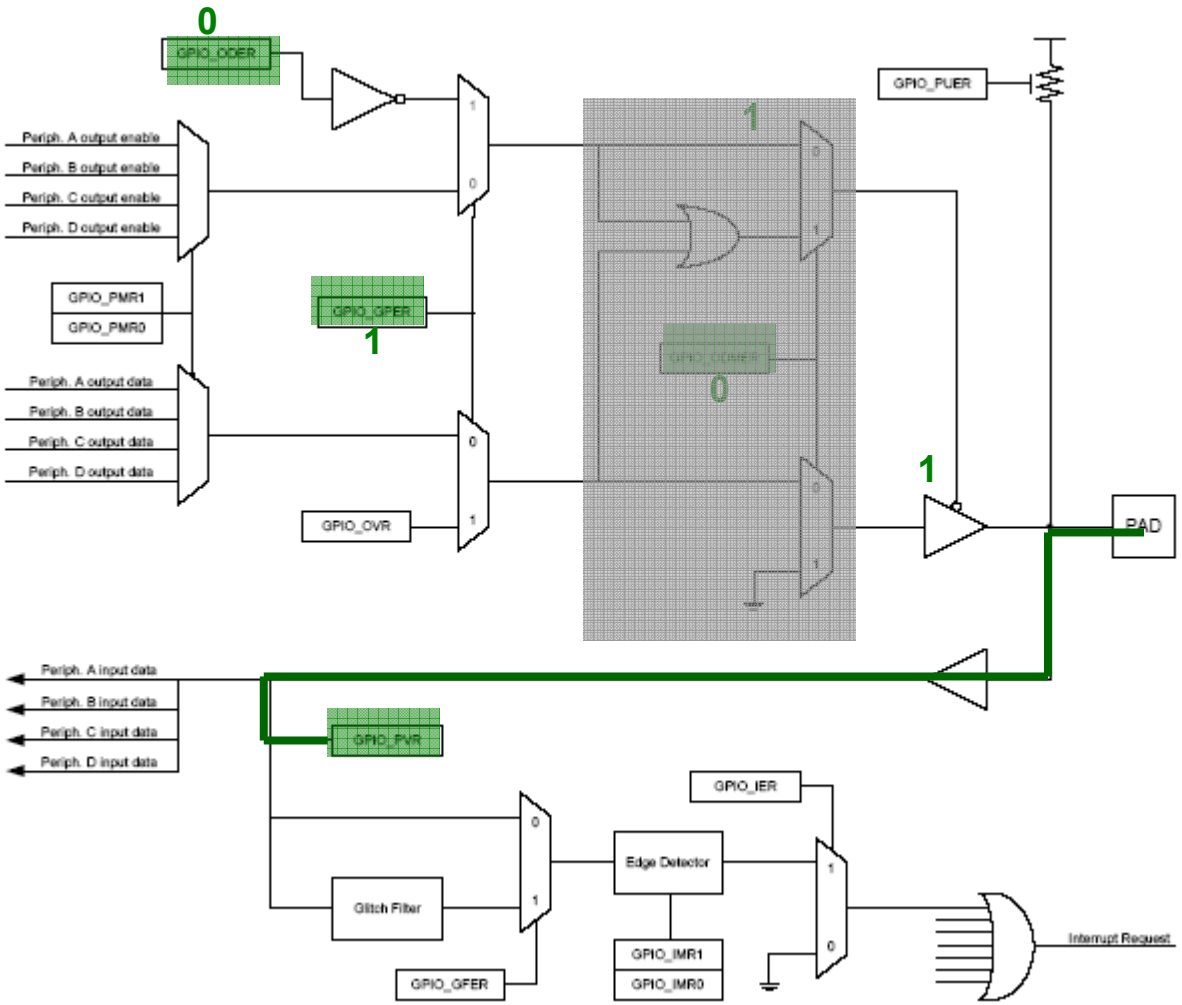


Κύτταρο GPIO (καταχωρητές 1-bit)



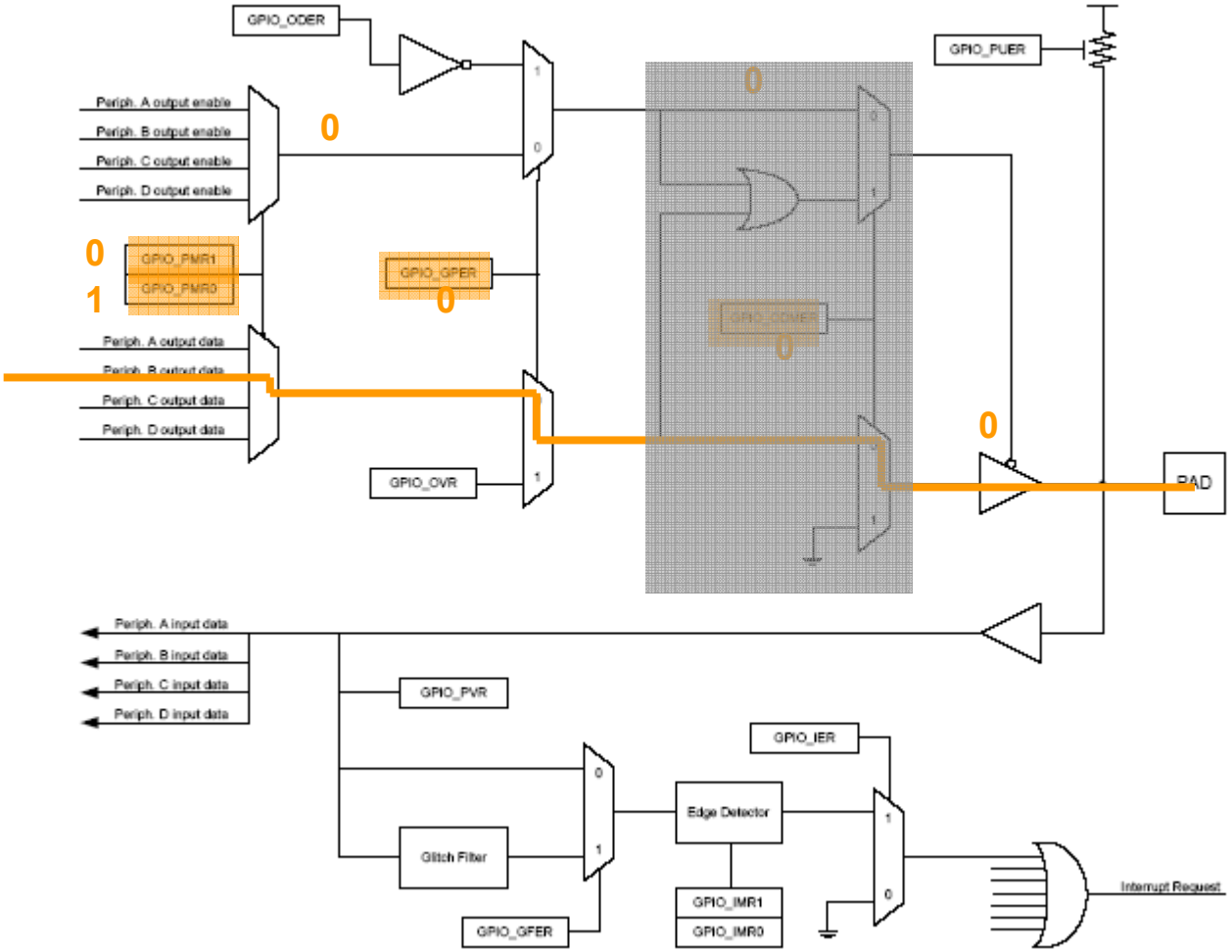
Απ' ευθείας έξοδος

Κύτταρο GPIO (καταχωρητές 1-bit)



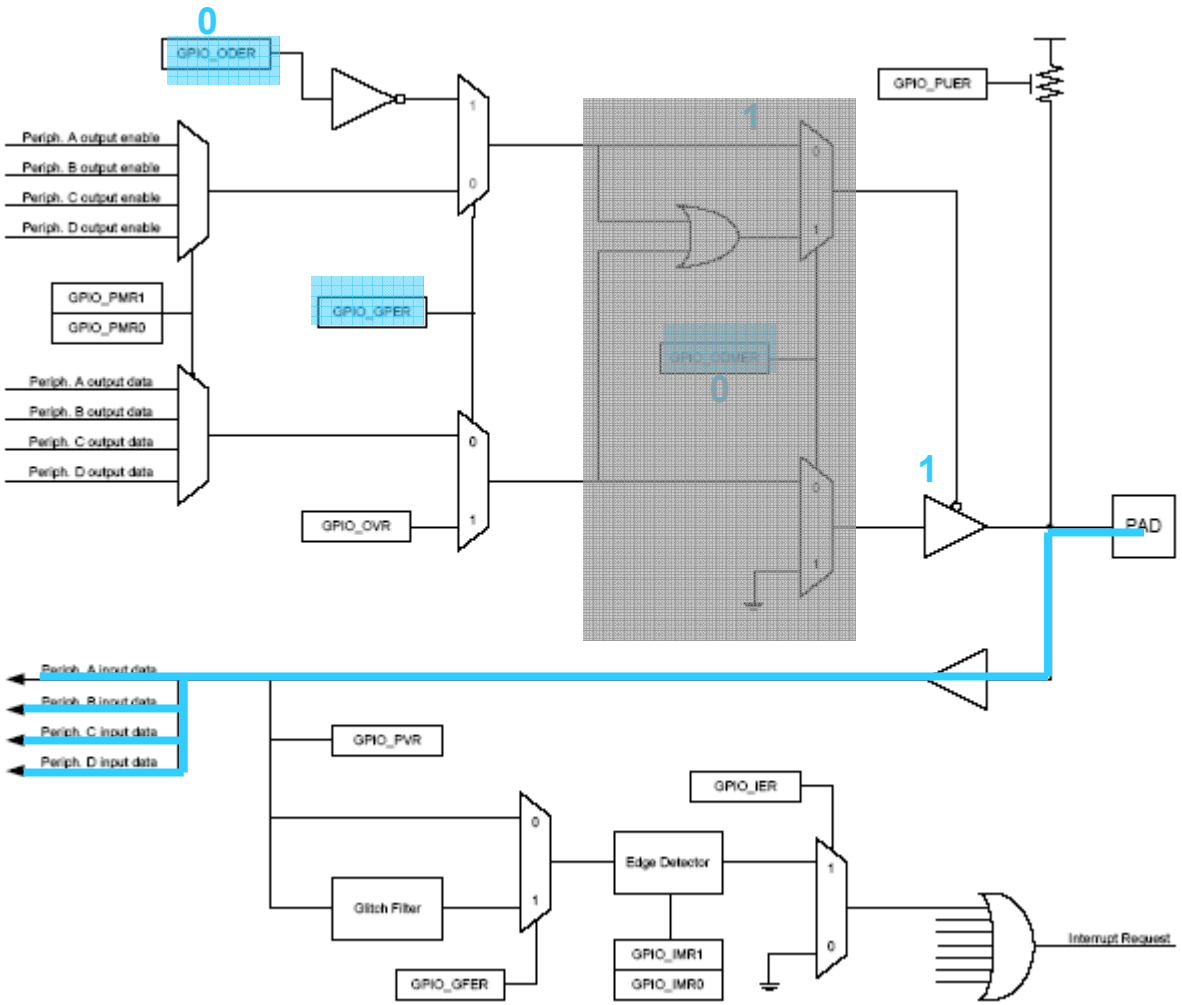
Απ' ευθείας είσοδος

Κύτταρο GPIO (καταχωρητές 1-bit)



Έξοδος από τα εσωτερικά περιφερειακά

Κύτταρο GPIO (καταχωρητές 1-bit)



Είσοδος προς τα εσωτερικά περιφερειακά

Προσπέλαση καταχωρητών ελέγχου GPIO

Αντιστοίχιση ακροδέκτη εξόδου (PIN) σε εσωτερικό ακροδέκτη (GPIO Pin)

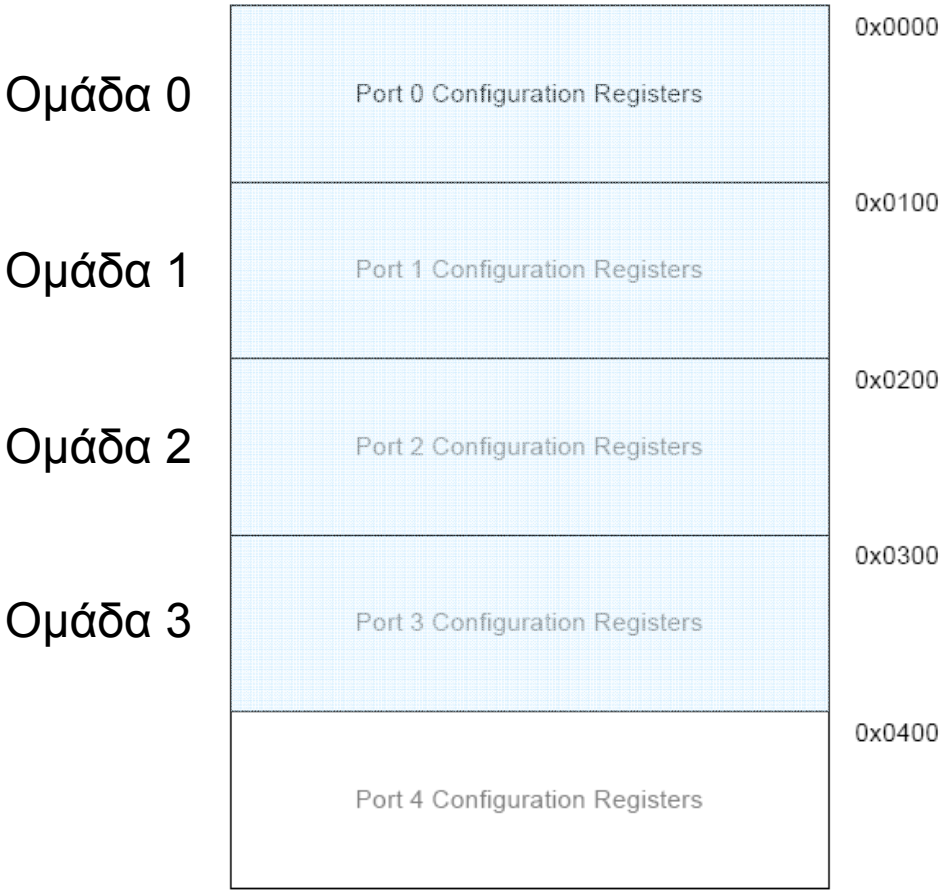
Table 12-9. GPIO Controller Function Multiplexing

TQFP100	VQFP144	PIN	GPIO Pin	Function A	Function B	Function C
19	25	PA00	GPIO 0	USART0 - RXD	TC - CLK0	
20	27	PA01	GPIO 1	USART0 - TXD	TC - CLK1	
23	30	PA02	GPIO 2	USART0 - CLK	TC - CLK2	
24	32	PA03	GPIO 3	USART0 - RTS	EIM - EXTINT[4]	DAC - DATA[0]
25	34	PA04	GPIO 4	USART0 - CTS	EIM - EXTINT[5]	DAC - DATAN[0]
26	39	PA05	GPIO 5	USART1 - RXD	PWM - PWM[4]	
27	41	PA06	GPIO 6	USART1 - TXD	PWM - PWM[5]	
28	43	PA07	GPIO 7	USART1 - CLK	PM - GCLK[0]	SPI0 - NPCS[3]
29	45	PA08	GPIO 8	USART1 - RTS	SPI0 - NPCS[1]	EIM - EXTINT[7]
30	47	PA09	GPIO 9	USART1 - CTS	SPI0 - NPCS[2]	MACB - WOL
31	48	PA10	GPIO 10	SPI0 - NPCS[0]	EIM - EXTINT[6]	
33	50	PA11	GPIO 11	SPI0 - MISO	USB - USB_ID	
...						
	135	PX35	GPIO 105	EBI - DATA[15]	SPI1 - MOSI	
	137	PX36	GPIO 104	EBI - DATA[14]	SPI1 - SCK	
	140	PX37	GPIO 103	EBI - DATA[13]	SPI1 - NPCS[0]	
	142	PX38	GPIO 102	EBI - DATA[12]	SPI1 - NPCS[1]	
	144	PX39	GPIO 101	EBI - DATA[11]	SPI1 - NPCS[2]	

Προσπέλαση καταχωρητών ελέγχου GPIO

Ομαδοποίηση καταχωρητών κυττάρου GPIO σε καταχωρητές 32 bits

101 ακροδέκτες GPIO χωρίζονται 4 ομάδες (ports) των 32 bits

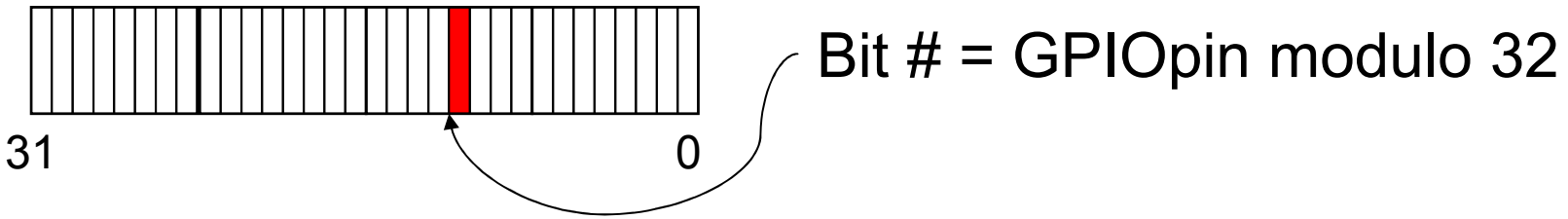


Σχετικές Διευθύνσεις ως προς τη Διεύθυνση Βάσης του GPIO (`0xFFFF1000`)

$$Port\# = \left\lfloor \frac{GPIOpin}{32} \right\rfloor$$

Προσπέλαση καταχωρητών ελέγχου GPIO

Καταχωρητής ελέγχου GPIO



0x40	Output Driver Enable Register	Read/Write	ODER	Read/Write	0x00000000
0x44	Output Driver Enable Register	Set	ODERS	Write-Only	
0x48	Output Driver Enable Register	Clear	ODERC	Write-Only	
0x4C	Output Driver Enable Register	Toggle	ODERT	Write-Only	
0x50	Output Value Register	Read/Write	OVR	Read/Write	0x00000000
0x54	Output Value Register	Set	OVRS	Write-Only	
0x58	Output Value Register	Clear	OVRC	Write-Only	
0x5c	Output Value Register	Toggle	OVRT	Write-Only	
0x60	Pin Value Register	Read	PVR	Read-Only	depending on pin states
0x64	Pin Value Register	-	-	-	
0x68	Pin Value Register	-	-	-	

Παράδειγμα για απ' ευθείας έξοδο

Ο ακροδέκτης **PB29** να τεθεί σε λογικό «1»

Ο ακροδέκτης PB29 αντιστοιχεί στον ακροδέκτη **GPIO 61**
[61 / 32] = 1 , δηλαδή αντιστοιχεί στην **GPIO port 1**
61 mod 32 = 29, δηλαδή το bit 29 επηρρεάζει τις λειτουργίες του PB29

Επομένως, το bit 29 του καταχωρητή **OVR** (Output Value Register) της GPIO Port 1 θα πρέπει να τεθεί σε λογικό «1»

Διεύθυνση Βάσης GPIO	`0xFFFF1000`
Σχετική Διεύθυνση GPIO port 1	`0x00000100`
Σχετική Διεύθυνση OVRS	`0x00000054`
Απόλυτη Διεύθυνση OVRS	**`0xFFFF1154`**

Τελικά, στην απόλυτη διεύθυνση **0xFFFF1154** πρέπει να γραφεί η λέξη `0x20000000` που αντιστοιχεί στην 0010 0000 0000 0000 0000 0000 0000 0000 σε δυαδικό σύστημα

Ανάλογοι υπολογισμοί θα πρέπει να γίνουν και για τους υπόλοιπους καταχωρητές GPIO που θα πρέπει να επηρρεαστούν (ODER κ.λ.π.)

Παράδειγμα γλώσσας C για απ' ευθείας έξοδο

Ο κώδικας C που επιτυγχάνει το ίδιο αποτέλεσμα !

```
#include "gpio.h"

gpio_set_gpio_pin(AVR32_PIN_PB29);
```

Η συνάρτηση στο gpio.c

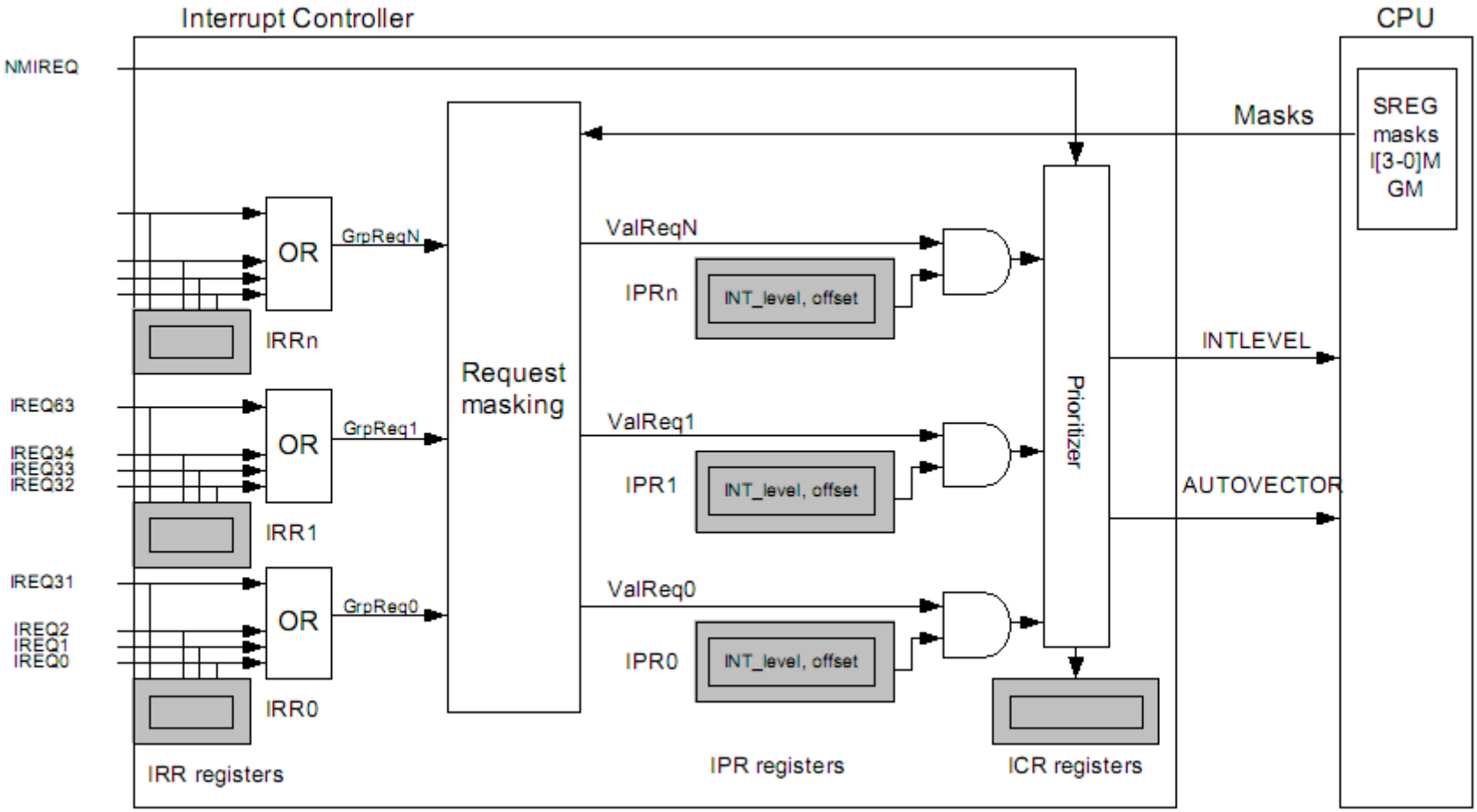
$$Port\# = \left\lfloor \frac{GPIOpin}{32} \right\rfloor$$

```
void gpio_set_gpio_pin(unsigned int pin)
{
        volatile avr32_gpio_port_t *gpio_port = &GPIO.port[pin >> 5];

        gpio_port->ovrs  = 1 << (pin & 0x1F);
        gpio_port->oders = 1 << (pin & 0x1F);
        gpio_port->gpers = 1 << (pin & 0x1F);
}
```

Bit # = GPIOpin modulo 32

Χειριστής Διακοπών



Πίνακας Απεικόνισης Διακοπών

Table 12-3. Interrupt Request Signal Map

Group	Line	Module	Signal
0	0	AVR32 UC CPU with optional MPU and optional OCD	SYSBLOCK COMPARE
	0	External Interrupt Controller	EIC 0
	1	External Interrupt Controller	EIC 1
.....			
2	4	General Purpose Input/Output	GPIO 4
	5	General Purpose Input/Output	GPIO 5
	6	General Purpose Input/Output	GPIO 6
	7	General Purpose Input/Output	GPIO 7
	8	General Purpose Input/Output	GPIO 8
.....			
10	0	Serial Peripheral Interface	SPI
11	0	Two-wire Interface	TWI
12	0	Pulse Width Modulation Controller	PWM
13	0	Synchronous Serial Controller	SSC
14	0	Timer/Counter	TC0
	1	Timer/Counter	TC1
	2	Timer/Counter	TC2
15	0	Analog to Digital Converter	ADC
16	0	Ethernet MAC	MACB
17	0	USB 2.0 OTG Interface	USBB
18	0	SDRAM Controller	SDRAMC
19	0	Audio Bitstream DAC	DAC

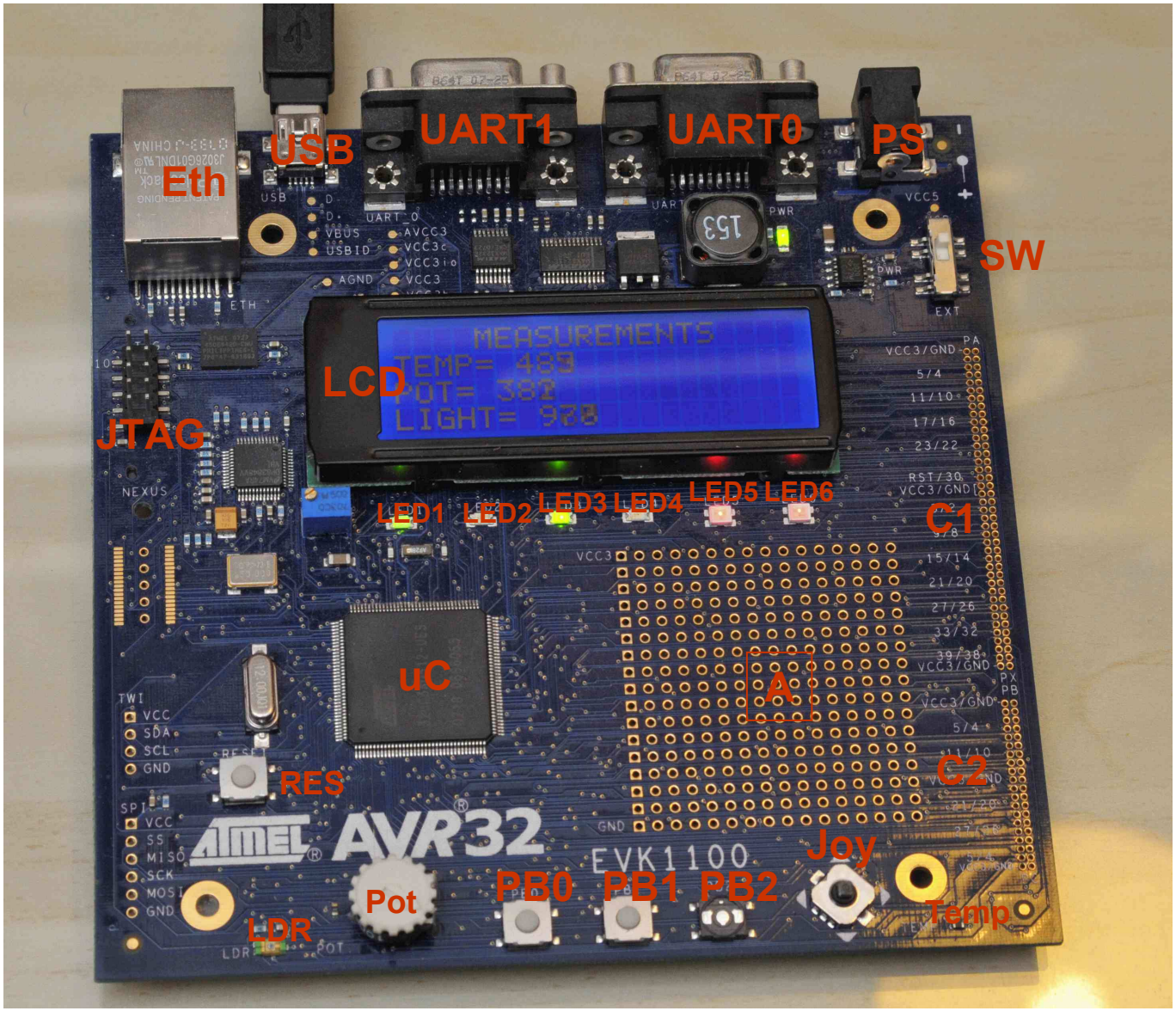
Παράδειγμα γλώσσας C για χειρισμό διακοπής

```
#include "intc.h"

__attribute__((__interrupt__))
static void PB0_int_handler(void)
{
... ...
gpio_clear_pin_interrupt_flag(GPIO_PUSH_BUTTON_0);
}

int main(void)
{
... ...
Disable_global_interrupt();
INTC_init_interrupts();
gpio_enable_pin_glitch_filter(GPIO_PUSH_BUTTON_0);
gpio_enable_pin_interrupt (GPIO_PUSH_BUTTON_0,GPIO_PIN_CHANGE);
INTC_register_interrupt(&PB0_int_handler,AVR32_GPIO_IRQ_11, AVR32_INTC_INT0);
Enable_global_interrupt();
... ...
}
```

Πλακέτα δοκιμών EVK1100



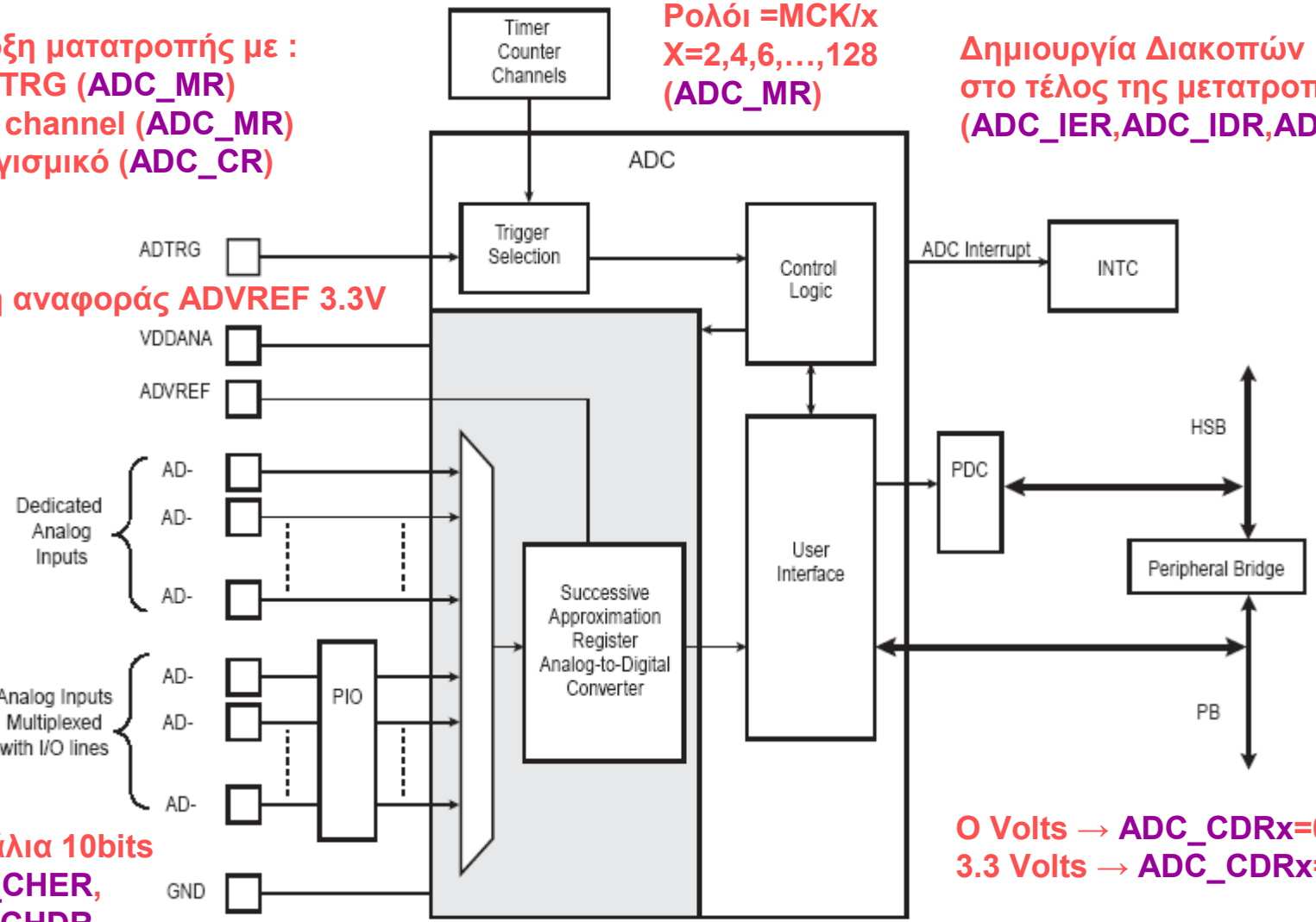
Προγραμματιστής JTAGICE mkII

Ανάπτυξη εφαρμογής με ADC
Γενική άποψη ADC

Έναρξη ματατροπής με :
1. ADTRG (ADC_MR)
2. TC channel (ADC_MR)
3. Λογισμικό (ADC_CR)

Ρολόι =MCK/x
X=2,4,6,…,128
(ADC_MR)

Δημιουργία Διακοπών
στο τέλος της μετατροπής
(ADC_IER,ADC_IDR,ADC_IMR)



Τάση αναφοράς ADVREF 3.3V

8 κανάλια 10bits
(ADC_CHER,
ADC_CHDR,
ADC_CHSR)

O Volts → ADC_CDRx=0
3.3 Volts → ADC_CDRx=1023

Ανάπτυξη εφαρμογής με ADC Καταχωρητές

33.7.1 ADC Control Register

Register Name: ADC_CR

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	START	SWRST

33.7.2 ADC Mode Register

Register Name: ADC_MR

Access Type: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	SHTIM			
23	22	21	20	19	18	17	16
–	–	–	STARTUP				
15	14	13	12	11	10	9	8
–	–	PRESCAL					
7	6	5	4	3	2	1	0
–	–	SLEEP	LOWRES	TRGSEL			TRGEN

Ανάπτυξη εφαρμογής με ADC Καταχωρητές

33.7.3 ADC Channel Enable Register

Register Name: ADC_CHER

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0

Επι πλέον : ADC_CHDR (Write Only) και ADC_CHSR (Read only)

Ανάπτυξη εφαρμογής με ADC
Καταχωρητές

33.7.6 ADC Status Register

Register Name: ADC_SR

Access Type: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RXBUFF	ENDRX	GOVRE	DRDY
15	14	13	12	11	10	9	8
OVRE7	OVRE6	OVRE5	OVRE4	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

EOCx : 1 για τέλος μετατροπής
OVRx : 1 για «χάσιμο» μετατροπής
DRDY : Το λογικό «Η» των EOCx
GOVR : Το λογικό «Η» των OVRx

Ανάπτυξη εφαρμογής με ADC Καταχωρητές

33.7.8 ADC Interrupt Enable Register

Register Name: ADC_IER

Access Type: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RXBUFF	ENDRX	GOVRE	DRDY
15	14	13	12	11	10	9	8
OVRE7	OVRE6	OVRE5	OVRE4	OVRE3	OVRE2	OVRE1	OVRE0
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

Επι πλέον : ADC_IDR (Write Only) και ADC_IMR (Read only)

33.7.11 ADC Channel Data Register

Register Name: ADC_CDRx

Access Type: Read-only

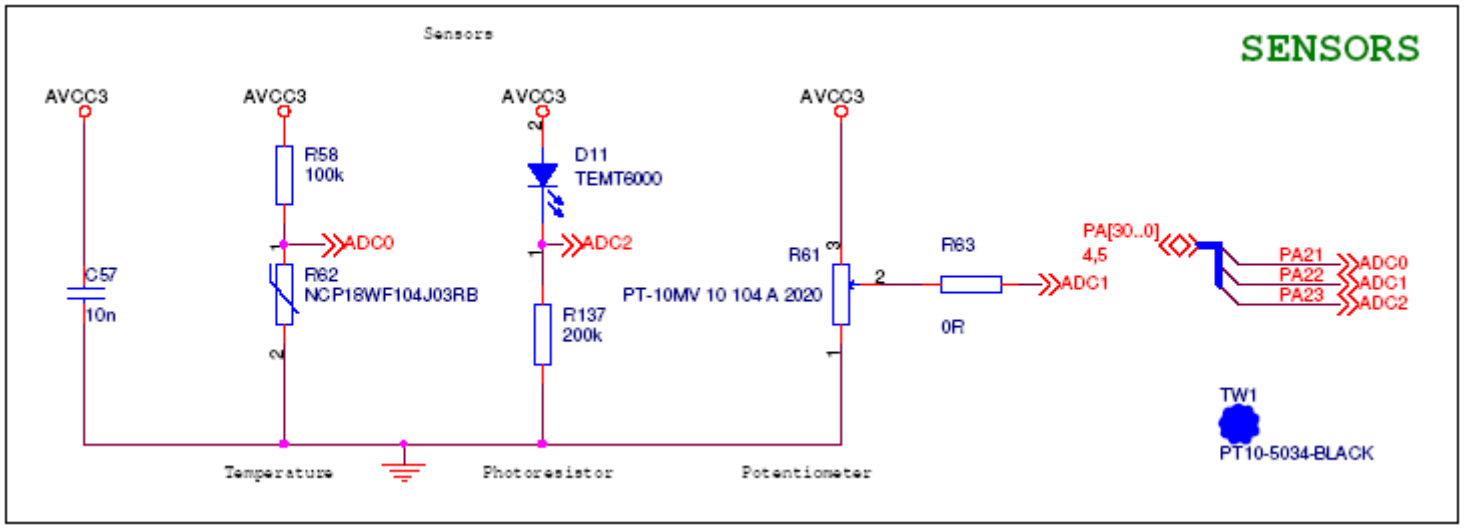
31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	DATA	
7	6	5	4	3	2	1	0
DATA							

Ανάπτυξη εφαρμογής με ADC
Σκοπός

ΚΥΡΙΩΣ ΠΡΟΓΡΑΜΜΑ
Αρχικοποιήσεις
While (1)
    Ξεκίνα μετατροπή στα εξής 3 κανάλια :
    θερμίστορ, φωτοντίστασης και ποτενσιομέτρου
    Καθυστέρηση
While end
ΤΕΛΟΣ

ΡΟΥΤΙΝΑ ΕΞΥΠΗΡΕΤΗΣΗΣ ΔΙΑΚΟΠΗΣ EOC
    Διάβασε τιμή θερμίστορ
    Ρύθμισε ανάλογα τη φωτεινότητα του LED
    Διάβασε τιμή φωτοαντίστασης
    Ρύθμισε ανάλογα τη φωτεινότητα του LED
    Διάβασε τιμή φωτοαντίστασης
    Ρύθμισε ανάλογα τη φωτεινότητα του LED
ΕΠΙΣΤΡΟΦΗ

Ανάπτυξη εφαρμογής με ADC
Κώδικας C - Αρχικοποιήσεις



Συνδέσεις καναλιών ADC 0,1 και 2 με τα pins PA21,PA22,PA23

```
gpio_enable_module_pin(AVR32_ADC_AD_0_PIN, AVR32_ADC_AD_0_FUNCTION);
gpio_enable_module_pin(AVR32_ADC_AD_1_PIN, AVR32_ADC_AD_1_FUNCTION);
gpio_enable_module_pin(AVR32_ADC_AD_2_PIN, AVR32_ADC_AD_2_FUNCTION);
```

PA21	GPIO 21	ADC - AD[0]	EIM - EXTINT[0]	USB - USB_ID
PA22	GPIO 22	ADC - AD[1]	EIM - EXTINT[1]	USB - USB_VBOF
PA23	GPIO 23	ADC - AD[2]	EIM - EXTINT[2]	DAC - DATA[1]

Ανάπτυξη εφαρμογής με ADC
Κώδικας C - Επικεφαλίδες

Διαδοχή ανεύρεσης αρχείων επικεφαλίδας (`.h`)

`adc.h → avr32/io.h → avr32/io.h → avr32/uc3a0512.h`

Κατάλογος τοποθεσίας

<Install_dir> /AVR32 Toolchain/avr32/include/avr32

Περιεχόμενο `avr32/uc3a0512.h`

```
...
/* ADC */
#define AVR32_ADC_NUM                       1

/* ADC */
#define AVR32_ADC_ADDRESS                   0xFFFF3C00
#define AVR32_ADC                           (*((volatile avr32_adc_t*)AVR32_ADC_ADDRESS))
#define AVR32_ADC_CLK_PBA                   68
#define AVR32_ADC_IRQ                       480
#define AVR32_ADC_ADC_CHANNELS_MSB          7
#define AVR32_ADC_ADC_DATA_MSB              9
#define AVR32_ADC_AD_0_PIN                  21
#define AVR32_ADC_AD_0_FUNCTION             0
#define AVR32_ADC_AD_1_PIN                  22
#define AVR32_ADC_AD_1_FUNCTION             0
#define AVR32_ADC_AD_2_PIN                  23
#define AVR32_ADC_AD_2_FUNCTION             0
...
```

Ανάπτυξη εφαρμογής με ADC
Κώδικας C - Αρχικοποιήσεις

```
// Ρύθμιση ADC
volatile avr32_adc_t *adc = &AVR32_ADC;

// set Sample/Hold time to max
adc->mr |= 0xF << AVR32_ADC_SHTIM_OFFSET;
// set Startup to max
adc->mr |= 0x1F << AVR32_ADC_STARTUP_OFFSET;
// Χρονισμός ADC = MCK/64
adc->mr |= 0x3F << AVR32_ADC_PRESCAL_OFFSET;
// Δημιουργία διακοπής από το τρίτο κανάλι
adc->ier |= 1 << AVR32_ADC_IER_EOC2_OFFSET;
// Ενεργοποίηση καναλιών 0,1,2
adc->cher = 7; //(1+2+4)
```

24

16

8

2

11 1111 γράφονται στον MR ολισθημένα κατά 8 θέσεις αριστερά

Ανάπτυξη εφαρμογής με ADC
Κώδικας C – Διακοπή από ADC

```
INTC_register_interrupt(&ADC_int_handler,AVR32_ADC_IRQ, AVR32_INTC_INT0);
Enable_global_interrupt();
```

480=15*32+0
Group 15
Line 0

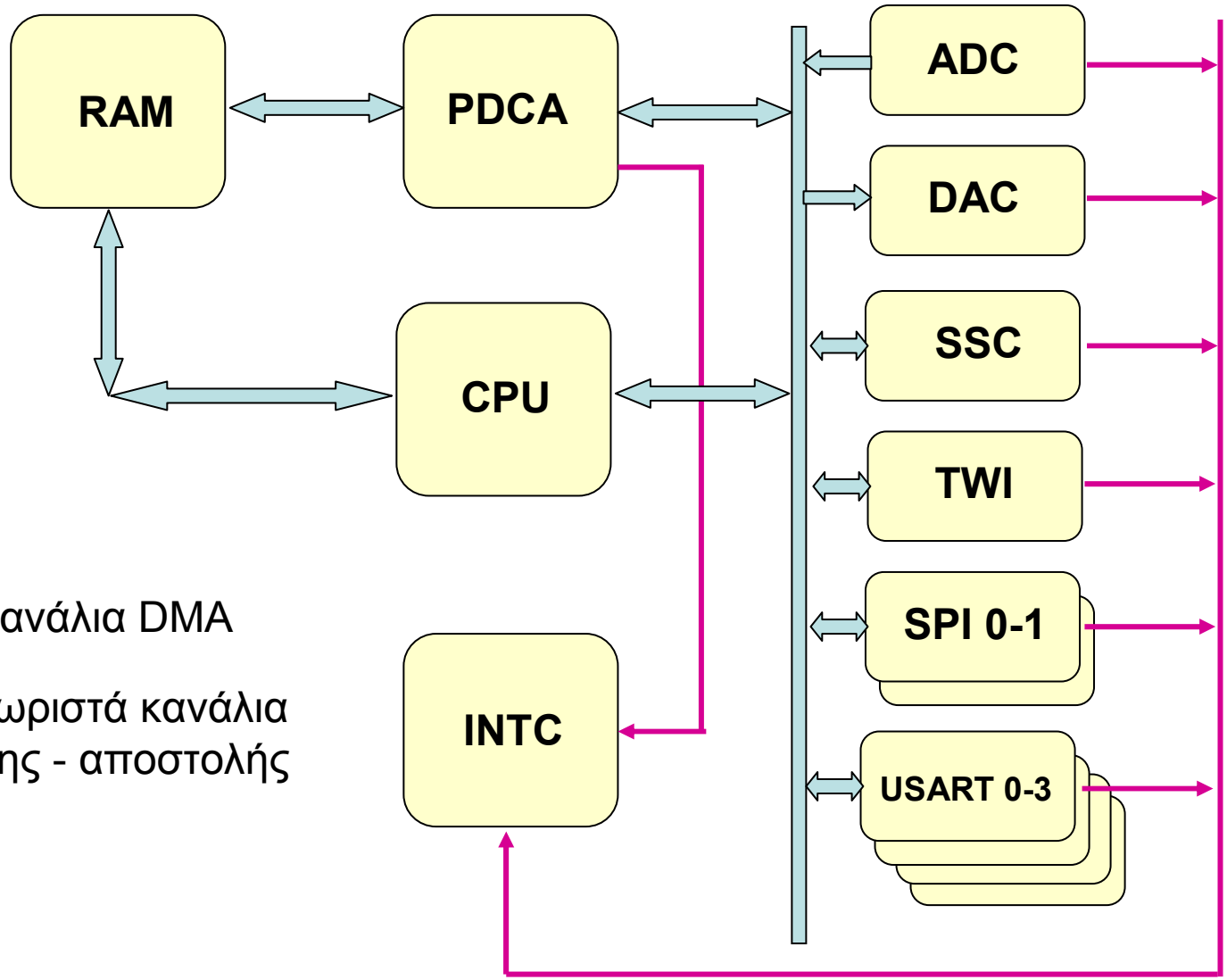
```
static void ADC_int_handler(void)
{
…
volatile avr32_adc_t *adc = &AVR32_ADC;
…

// Ανάγνωση ADC
tempr=adc_get_value(adc,0);
pot=adc_get_value(adc,1);
light=adc_get_value(adc,2);
…
}
```

Ανάπτυξη εφαρμογής με ADC
Κώδικας C – Κυρίως Πρόγραμμα

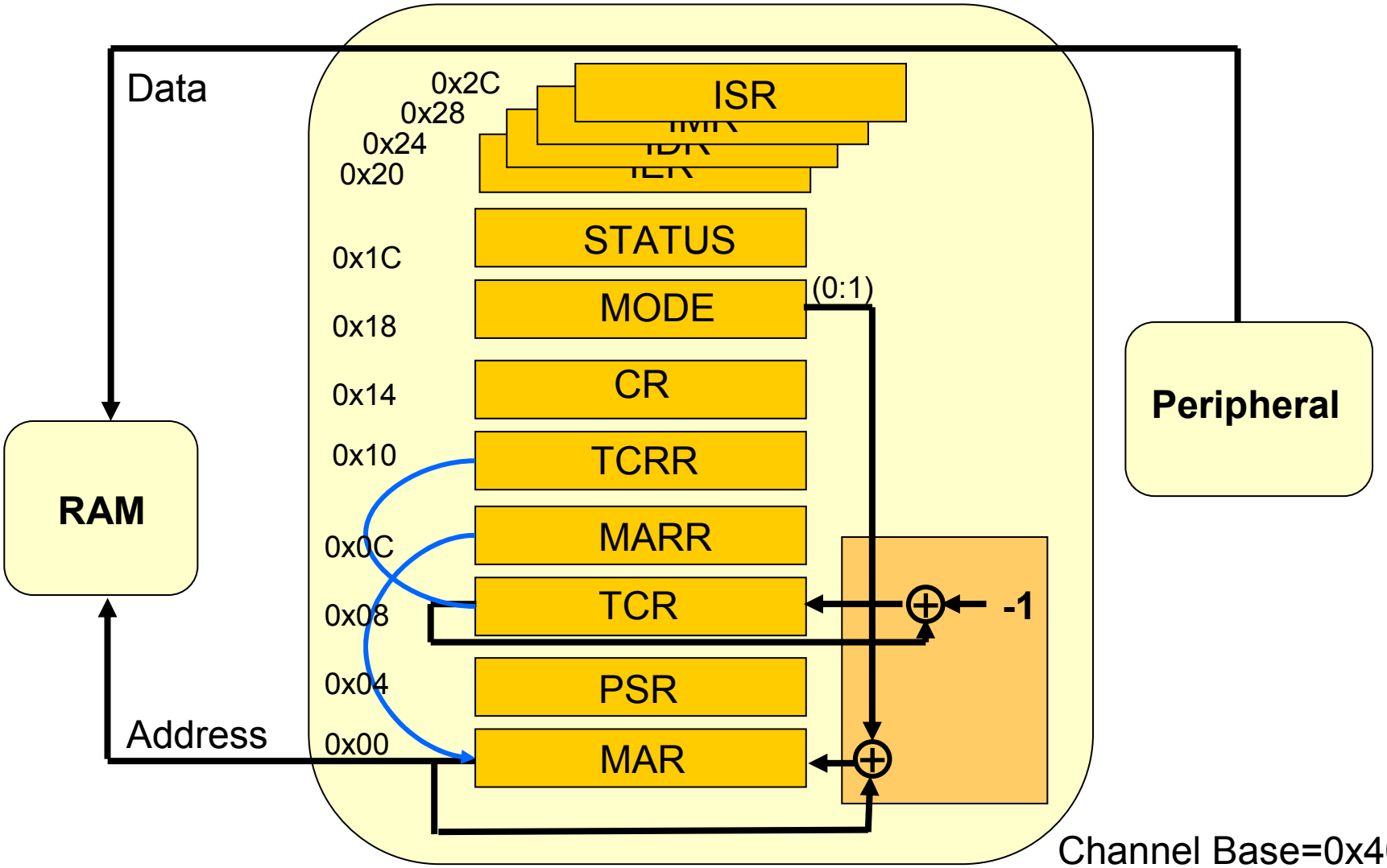
```
while (1)                                   // Ατέρμονος βρόχος
{
for (i = 0; i < 500; i += 1)                // Καθυστέρηση
adc_start(adc);                             // Έναρξη νέων μετρήσεων
gpio_tgl_gpio_pin(LED2_GPIO);               // Αναβόσημα LED3
}
```

Απ' ευθείας προσπέλαση στη μνήμη (DMA)



- 18 κανάλια DMA
- Ξεχωριστά κανάλια λήψης - αποστολής

Λειτουργία ενός καναλιού DMA

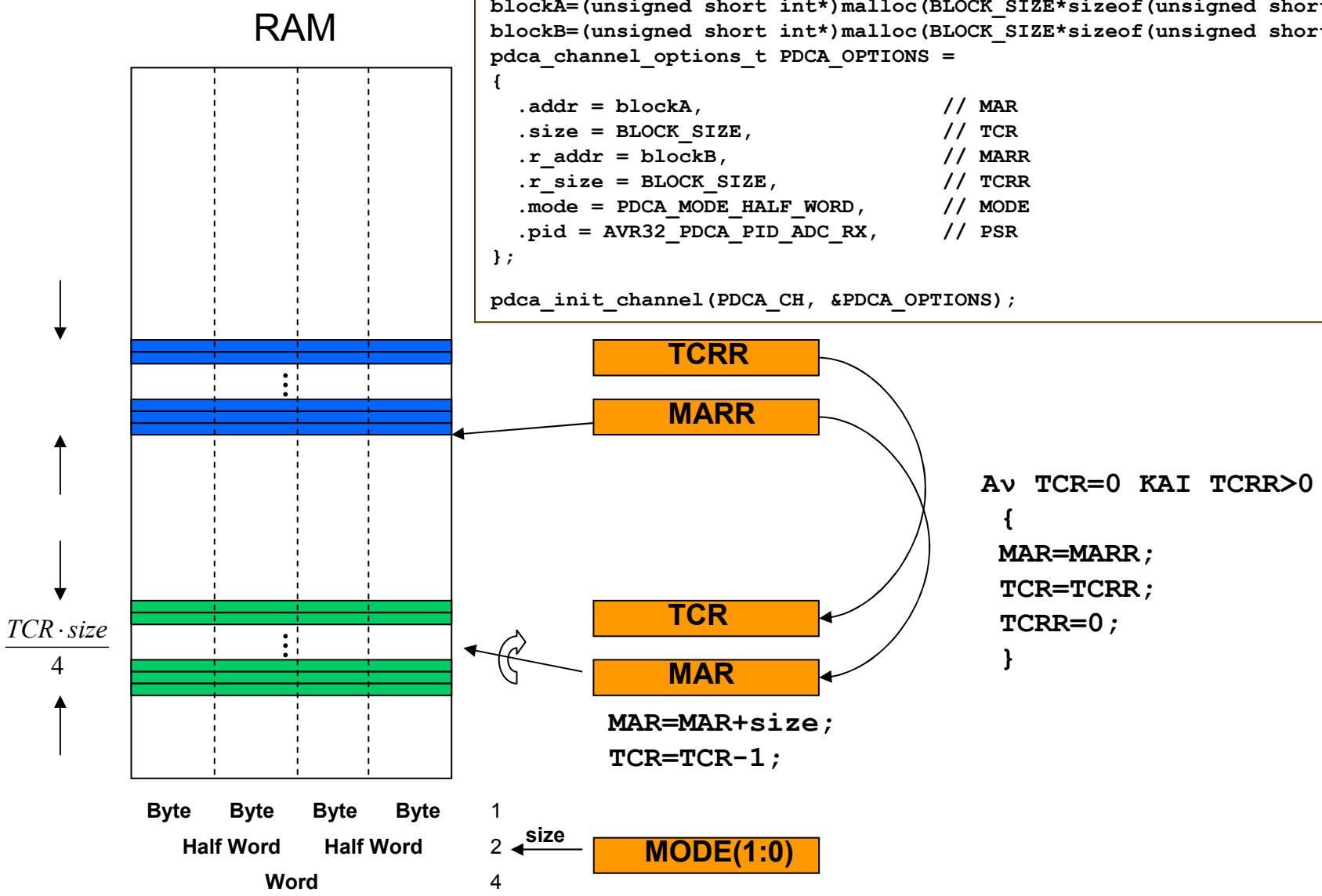


Register Address = PDCA Base + Channel Base + Register Offset

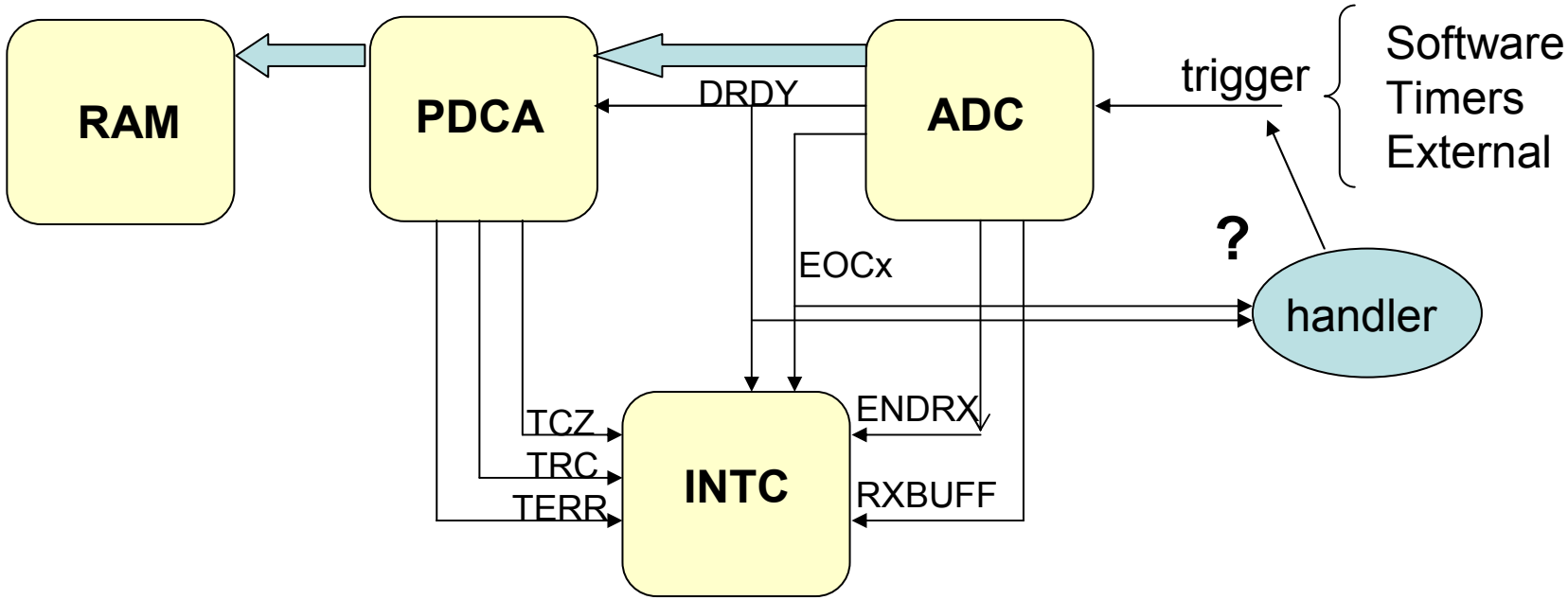
DMA - Εναλλαγή τμημάτων

```
blockA=(unsigned short int*)malloc(BLOCK_SIZE*sizeof(unsigned short int));
blockB=(unsigned short int*)malloc(BLOCK_SIZE*sizeof(unsigned short int));
pdca_channel_options_t PDCA_OPTIONS =
{
  .addr = blockA,                    // MAR
  .size = BLOCK_SIZE,                // TCR
  .r_addr = blockB,                  // MARR
  .r_size = BLOCK_SIZE,              // TCRR
  .mode = PDCA_MODE_HALF_WORD,       // MODE
  .pid = AVR32_PDCA_PID_ADC_RX,      // PSR
};

pdca_init_channel(PDCA_CH, &PDCA_OPTIONS);
```



DMA / ADC - Διακοπές



TCZ : Transfer Counter Zero (TCR=0)
TRC : Transfer Complete (TCR=0 Λ TCRR=0)
TERR : Transfer Error
ENDRX : End of Receive Buffer (TCR=0)
RXBUFF : Receive Buffer Full (TCR=0 Λ TCRR=0)

DMA – Συναρτήσεις C

- Ορισμός στα `pdca.h` και `pdca.c`
- Δομές δεδομένων, σταθερές στα `uc3a0512es.h` και `pdca100.h`
- Ένα σύνολο συναρτήσεων που καλύπτει εγγραφή σε όλους τους καταχωρητές :

```
int pdca_init_channel(unsigned int pdca_ch_number, const pdca_channel_options_t *opt)
```
Γράφει στους : `MAR, TCR, MARR, TCRR, MODE, PSR`

```
void pdca_enable_interrupt_reload_counter_zero(unsigned int pdca_ch_number)
```
Γράφει στους : `IER => bit RCZ`

```
void pdca_reload_channel(unsigned int pdca_ch_number, volatile void *addr, unsigned int size)
```
Γράφει στους : `MARR, TCRR`

```
void pdca_enable(unsigned int pdca_ch_number)
```
Γράφει στους : `CR => bit TEN`