

**Τμήμα Μηχανικών Η/Υ & Πληροφορικής**

**Εργαστήριο Προηγμένων Μικροεπεξεργαστών**

**Άσκηση 2**

|  |  |
| --- | --- |
| **Ονοματεπώνυμο** | **ΑΜ** |
| Εμμανουήλ Κατεφίδης | 6077 |
| Παναγιώτης Σταυρινάκης | 6217 |

**a) Κώδικας:**

|  |
| --- |
| #include <sys/types.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <sys/ioctl.h>  #include <unistd.h>  #include <sys/mman.h>  #include <stdio.h>  #include <stdlib.h>  #include "header.h"  PIO\* pioa = NULL;  AIC\* aic = NULL;  TC\* tc = NULL;  int main( int argc, const char\* argv[] ){  char tmp;  unsigned int i, counter, bitmask, next;  STARTUP; //ΑΡΧΙΚΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ  pioa -> PER = 0;  pioa -> OER = 0;  pioa -> CODR = 0;  for( i=0; i<7; i++ )  {  pioa -> PER |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΓΕΝΙΚΟΥ ΣΚΟΠΟΥ  pioa -> CODR |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΔΥΝΑΜΙΚΟ ΕΞΟΔΟΥ LOW  pioa -> OER |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΛΕΙΤΟΥΡΓΙΑ ΕΞΟΔΟΥ  }  counter = 100;  while( (tmp = getchar()) != 'e')  {  counter += 1;  if(counter >= 100)  {  pioa -> CODR = 0x7F; //ΚΑΘΑΡΙΣΜΟΣ ΟΛΩΝ ΤΩΝ OUTPUTS(LEDS)  bitmask = 1;  next = 20;  counter = 0;  }else{  if(counter >= next && counter <= 80)  {  pioa -> SODR = bitmask; //ΕΝΕΡΓΟΠΟΙΗΣΗ OUTPUT(LED)  bitmask <<= 1;  next += 10;  }  }  }  CLEANUP;  return 0;  } |

**b) Κώδικας:**

|  |
| --- |
| #include <sys/types.h>  #include <sys/stat.h>  #include <fcntl.h>  #include <sys/ioctl.h>  #include <unistd.h>  #include <sys/mman.h>  #include <stdio.h>  #include <stdlib.h>  #include "header.h"  #define PIOA\_ID 2  #define TC0\_ID 17  #define BUT\_IDLE 0  #define BUT\_PRESSED 1  #define BUT\_RELEASED 2  #define LED\_IDLE 0  #define LED\_LEFT 1  #define LED\_RIGHT 2  void FIQ\_handler(void);  PIO\* pioa = NULL;  AIC\* aic = NULL;  TC\* tc = NULL;  unsigned int button\_state = BUT\_IDLE;  unsigned int led\_state = LED\_IDLE;  int mask;  int main( int argc, const char\* argv[] )  {  char tmp;  unsigned int gen, i, counter, bitmask, next;  STARTUP; //ΑΡΧΙΚΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ  pioa -> PER = 0;  pioa -> OER = 0;  pioa -> CODR = 0;  for( i=0; i<7; i++ )  {  pioa -> PER |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΓΕΝΙΚΟΥ ΣΚΟΠΟΥ  pioa -> CODR |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΔΥΝΑΜΙΚΟ ΕΞΟΔΟΥ LOW  pioa -> OER |= (1)<<(i) ; //ΓΡΑΜΜΗ i: ΛΕΙΤΟΥΡΓΙΑ ΕΞΟΔΟΥ  }  tc->Channel\_0.RC = 1638; //8192(period of 1 sec) \* 0.2(5Hz = 0.2 sec) = 1638  tc->Channel\_0.CMR = 0x2084; //SLOW CLOCK, WAVEFORM, DISABLE CLK ON RC COMPARE  tc->Channel\_0.IDR = 0xFF; //ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ ΟΛΩΝ ΤΩΝ ΔΙΑΚΟΠΩΝ  tc->Channel\_0.IER = 0x10; //ΕΝΕΡΓΟΠΟΙΗΣΗ ΜΟΝΟ ΤΟΥ RC COMPARE  gen = tc->Channel\_0.SR; //TC0 : ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑΚΟΠΕΣ  pioa->PER |= (1)<<(7); //ΓΡΑΜΜΗ 7: ΓΕΝΙΚΟΥ ΣΚΟΠΟΥ  pioa->PUER = (1)<<(7); //ΕΝΕΡΓΟΠΟΙΗΣΗ ΣΤΗ ΓΡΑΜΜΗ 7: PULL−UP  pioa->ODR = (1)<<(7); //ΓΡΑΜΜΗ 7: ΛΕΙΤΟΥΡΓΙΑ ΕΙΣΟΔΟΥ  gen = pioa->ISR; // PIOA: ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑΚΟΠΕΣ  pioa->IER = (1)<<(7); //ΕΝΕΡΓΟΠΟΙΗΣΗ ΔΙΑΚΟΠΩΝ ΣΤΗ ΓΡΑΜΜΗ 7  aic->FFER = (1<<PIOA\_ID) | (1<<TC0\_ID); //ΟΙ ΔΙΑΚΟΠΕΣ 2 ,17 ΕΙΝΑΙ ΤΥΠΟΥ FIQ  aic->IECR = (1<<PIOA\_ID) | (1<<TC0\_ID); //ΕΝΕΡΓΟΠΟΙΗΣΗ ΔΙΑΚΟΠΩΝ : PIOA & TC0  aic->ICCR = (1<<PIOA\_ID)|(1<<TC0\_ID); //AIC : ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑΚΟΠΕΣ  counter = 100;  mask = 1;  while( (tmp = getchar()) != 'e')  {  counter += 1;  if(counter >= 100)  {  pioa -> CODR = 0x7F; //ΚΑΘΑΡΙΣΜΟΣ ΟΛΩΝ ΤΩΝ OUTPUTS(LEDS)  bitmask = mask;  next = 20;  counter = 0;  }else{  if(counter >= next && counter <= 80)  {  pioa -> SODR = bitmask; //ΕΝΕΡΓΟΠΟΙΗΣΗ OUTPUT(LED)  bitmask <<= 1;  if(bitmask == 0x80) bitmask = 1;  next += 10;  }  }  }  aic->IDCR = (1<<PIOA\_ID) | (1<<TC0\_ID); //ΔΙΑΚΟΠΗ ΤΩΝ AIC interrupts  tc->Channel\_0.CCR = 0x02; //ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ ΤΟΥ Timer  CLEANUP;  return 0;  }  void FIQ\_handler(void)  {  unsigned int data\_in = 0;  unsigned int fiq = 0;  unsigned int data\_out;  fiq = aic->IPR; //ΕΝΤΟΠΙΣΜΟΣ ΠΕΡΙΦΕΡΕΙΑΚΟΥ ΠΟΥ ΠΡΟΚΑΛΕΣΕ ΤΗ ΔΙΑΚΟΠΗ    if( fiq & (1<<PIOA\_ID) ) //ΕΛΕΓΧΟΣ ΓΙΑ PIOA  {  data\_in = pioa->ISR; //ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΠΗΓΗΣ ΤΗΣ ΔΙΑΚΟΠΗΣ  aic->ICCR = (1<<PIOA\_ID); //ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΔΙΑΚΟΠΗΣ ΑΠΟ AIC  data\_in = pioa->PDSR; //ΑΝΑΓΝΩΣΗ ΤΙΜΩΝ ΕΙΣΟΔΟΥ  if( data\_in & 0x80 ) //ΔΙΑΚΟΠΤΗΣ ΠΑΤΗΜΕΝΟΣ  {  if(button\_state == BUT\_IDLE)  {  button\_state = BUT\_PRESSED;  if( led\_state == LED\_IDLE ) //ΑΝ ΔΕΝ ΑΝΑΒΟΣΒΗΝΕΙ  {  led\_state = LED\_RIGHT;  tc->Channel\_0.CCR = 0x05; //ΕΝΑΡΞΗ ΜΕΤΡΗΤΗ  }else if( led\_state == LED\_RIGHT )  {  led\_state = LED\_LEFT;  tc->Channel\_0.CCR = 0x05; //ΕΝΑΡΞΗ ΜΕΤΡΗΤΗ  }else{  led\_state = LED\_IDLE;  tc->Channel\_0.CCR = 0x02; //ΔΙΑΚΟΠΗ ΜΕΤΡΗΤΗ  }  }  }else{  if(button\_state == BUT\_PRESSED) button\_state = BUT\_IDLE;  }  }    if( fiq & (1<<TC0\_ID) )  {  data\_out = tc->Channel\_0.SR;//ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΠΗΓΗΣ ΤΗΣ ΔΙΑΚΟΠΗΣ  aic->ICCR = (1<<TC0\_ID); //ΕΚΚΑΘΑΡΙΣΗ ΔΙΑΚΟΠΗΣ ΚΑΙ ΑΠΟ AIC  if(led\_state == LED\_RIGHT)  {  mask >>= 1;  if(mask == 0x00) mask = 0x40;  }else if( led\_state == LED\_LEFT )  {  mask <<= 1;  if(mask == 0x80) mask = 1;  }  tc->Channel\_0.CCR = 0x05; //ΕΝΑΡΞΗ ΜΕΤΡΗΤΗ  }  } |