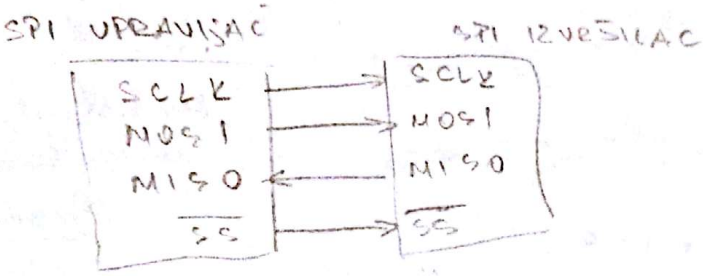
Lj R 2015/2016.

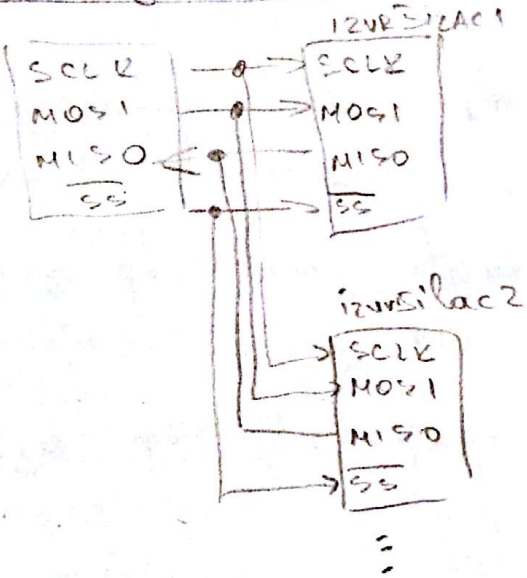
① Serial Peripheral Interface

- serijska, sinkrona, dvosmjerna sabirnica

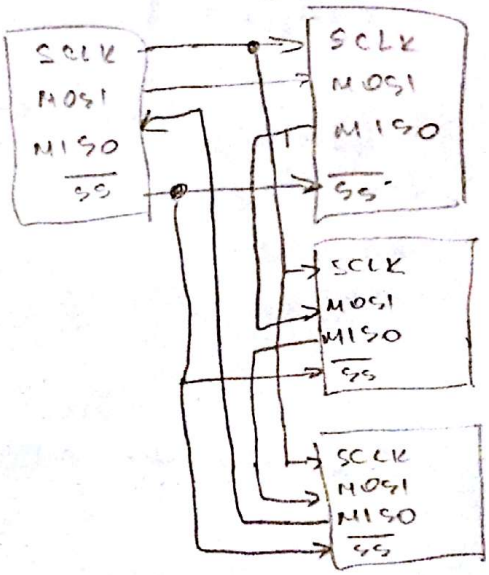
1 izvršioc



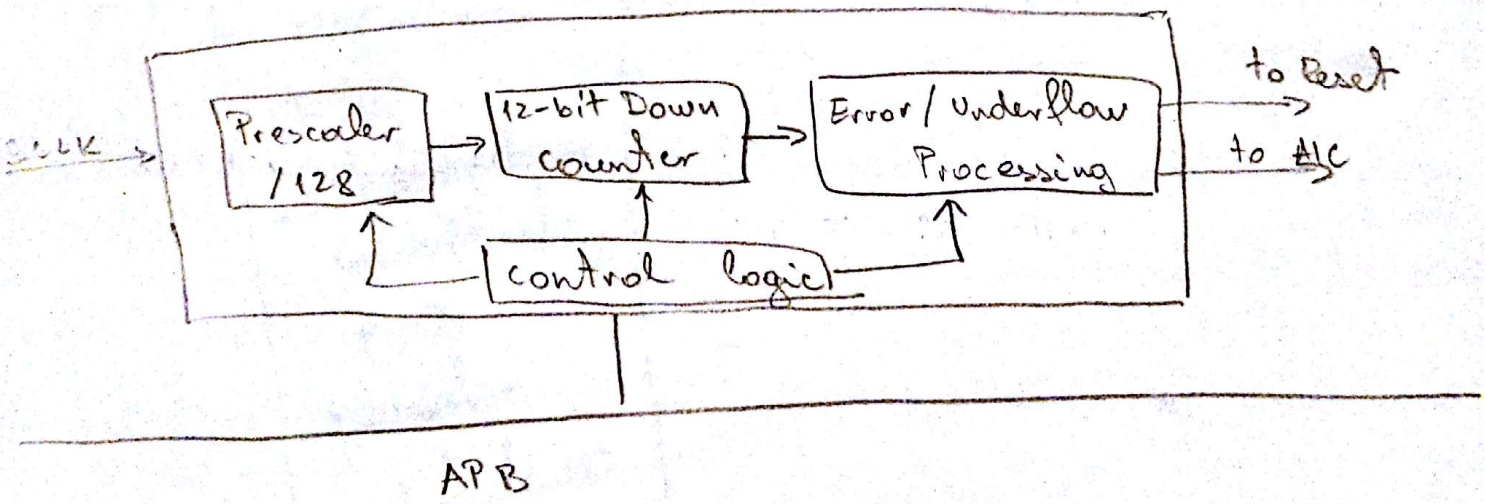
više od jednog izvršioca - Paralelno



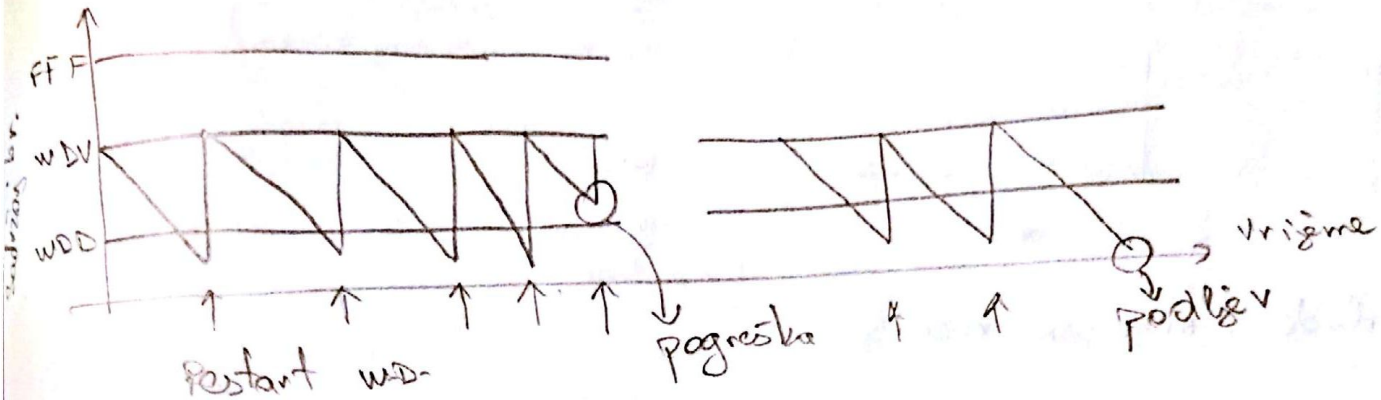
lanac



② WATCH DOG



Princip rada



- brojilo- broji od vrijednosti WDV (watch dog value) prema 0
- kad odbroji do 0 -> podšev (Underflow)
- kad se brojilo restarta, a sadržaj mu niže manji od vrijednosti: WDD (watch dog delta) -> pogreška (Error)
- oba uzrokuju reset ; ili prekid ako su omogućeni

(3.)

if == "P" -> izvor takta PLL 40MHz 20 u stac

-malo utitravanja

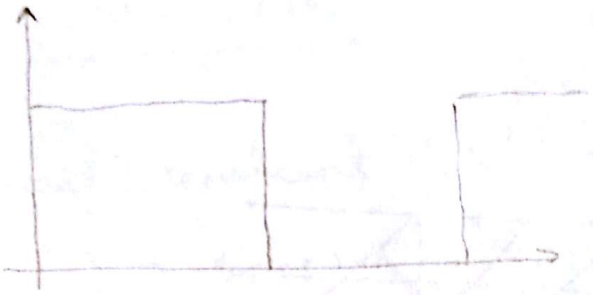
$$f_{PLLCK} = f_{MAINCK} \frac{MUL+1}{DIV \cdot 2^{PRES}}$$

~~30~~ $= 20 \frac{1}{1} = 80$

```
#include <stdio.h>
#include < AT91SAM7X512.h>

int main(void) {
    }

void promjena (char slovo){
    if(slovo == "P")
* AT91C_CKGR_PLLR = (3<<16)|(0b00<<14)|(0x3F<<8)|(1<<0)
while ((*AT91C_PMC_SR & (1<<2)==0);

* AT91C_PMC_MCKR = (1<<2) /-> prescaler = 1
while ((*AT91C_PMC_SR & (1<<3)==0);

* AT91C_PMC_MCKR = (3<<0);
while((*AT91C_PMC_SR &(1<<3)==0);
  }
}
```

; DIV = 1
; PLLCOUT = 0x3F (63)zbs
/MUL = 3, OUT = 0b00

4.



10 Hz

PIO 2 - signal

PIO 3 - komplement

PIT

32 MHz

```
# include <AT91SAM7X512.h>

void sys_handler (void) __irq;
void spurious_handler (void) __irq;

unsigned int signal = 0;
unsigned int brojac = 0;

int main (void) {
    *(AT91C_AIC_SMR + AT91C_ID_SYS) = (0x0 << 0) | (0x1 << 5);
    *(AT91C_AIC_SVR + AT91C_ID_SYS) = (unsigned int) sys_handler;
    *(AT91C_AIC_SPU) = (unsigned int) spurious_handler;
    *(AT91C_AIC_IECR) = 1 << AT91C_ID_SYS;
    // Upis i omogućavanje u PIT, ima prescaler /16 nad uređajem
    // 32MHz - takt, -> radi na 2MHz a potrebno je 10Hz
    // potrebno da broji do 2M/10 = 200000 -> trajanje = 20000
    *AT91C_PITC_PIMR = AT91C_PITC_PITIEN | AT91C_PITC_PITEN |
    *AT91C_PMC_PCER = (trajanje & 0xFFFFF);
                      (1 << AT91C_ID_PIOA);
    *AT91C_PIOA_PER = 0x1 << 2 | 0x1 << 3;
    *AT91C_PIOA_OER = 0x1 << 2 | 0x1 << 3;
```

```
while (1) {}

void sys_handler (void) __irq{
  brojac ++;
  if (brojac % 20=0)
    {
      signal = signal ^ 1;  XOR 1, x
      brojac=0
    }
  if (signal ==0){
      *AT91C_PIOA_CODR = 0x1<<2;
       //      SODR = 0x1<<2;
  else {
      *AT91C_PIOA_SODR = 0x1<<2;
      *      //      CODR = 0x1<<2;
  *AT91C_AIC_EOICR = *AT91C_PITC_PIVR;
  }

void spurious_handler (void) __irq {
    *AT91C_AIC_EOICR=0;
  }
```

5.
```
void c_zadatak (int arg0, int arg1, int * args){
  char * adresa = (char *)(*(args+5));  // PO-P4
  int povratak1 =*(args+6);
  int povratak2 =*(args+7);
  char poruka[5]
  strcpy( poruka," Reset");
  if (strcmp( poruka, adresa)==0)
  *(args+8) = povratak1;
  else { *(args+8) = povratak2;}
  }
  return;
```

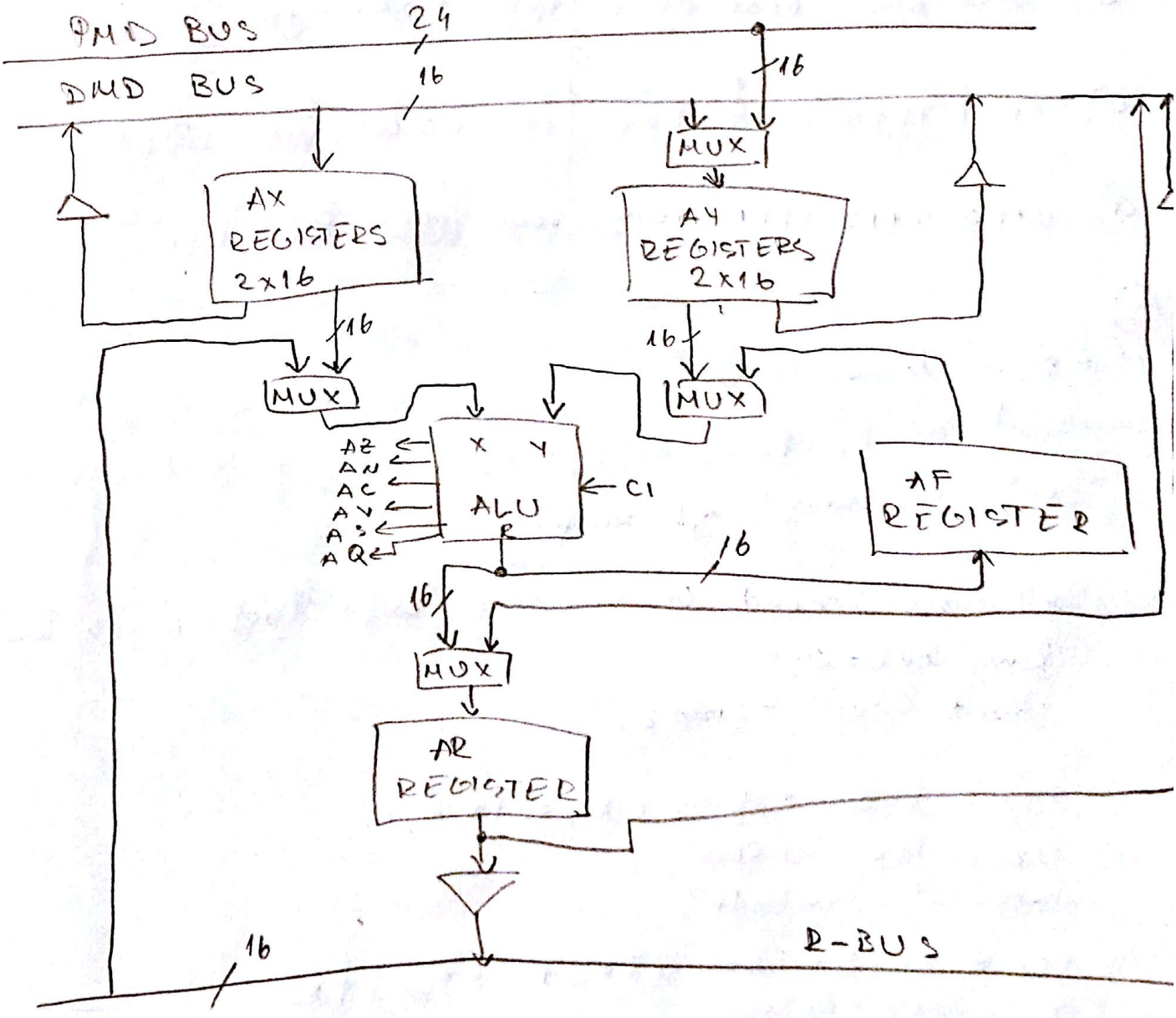
(6)

```
int main (void) {

 char  znak, proslizmak;
 ini t_PIOB ();
 scanf ("%c", &znak);
 printf ("%c", &znak);
proslizmak = znak;
 while (1) {

    if ( scanf (%c", &znak)!=proslizmak){
                printf("%c", znak);
                proslizmak = znak;
                }

        }
```

```
# include <stdio.h>
# include <AT91SAM7X512.h>
# pragma import (use_no_semihosting_swi)

struct __FILE{
    int handle;
   };
FILE __stdin, __stdout, __stderr;

int fputc (int ch, FILE *f){
    char slovo = ch;
      *AT91C_PIOB_SODR = (~slovo)<<8;
      return ch;
      }
in ferror (FILE *f) {
    return 0;
  }
```

```
int fgetc (FILE *f){
    char c = *ATGIC_PIOB_PDSR & 0xFF;
    return c;
}

int __backspace (){
    return 0;
}

void _sys_exit (int return_code){
    while (1);
}
```

(7) ALU jedinica



(8)

a) SR = LSHIFT S1 BY -4 (LO);

b) SR = LSHIFT S1 BY 3 (LO);

c) SR = ASHIFT S1 BY -6 (HI);

d) SR = ASHIFT S1 BY 1 (HI);

S1 = 0x B37A → lijevo gdjvice desno nule

1011 0011 0111 1010

	S1	S0
a)	0000 0000 0000 0000	0000 1011 0011 0111
b)	0000 0000 0000 0101	1001 1011 1010 0000
c)	1111 1110 1100 1101	1110 1000 0000 0000
d)	0110 0110 1111 0100	0000 0000 0000 0000

(9.)

```
#define Nsamp 1000
short a0, k, a1[2], a2[2], b1[2], b2[2]
short x[Nsamp], y[Nsamp]

short proces second order (short x[n], koef1, koef2, koef3, koef4){
    long rez = 0;
    short novo_st; izlaz;

    rez = koef3 * s2 * 2 + b2 * s1 * 2;
    rez = rez + 0x8000;
    novo_st = rez >> 16;

    rez = xn * 2 + s2 * koef1 * 2 + s3 * koef1 * 2
    rez = rez + 0x8000;
    novo_st += rez >> 16;
    s2 = s1; s1 = xn; s3 = s2; s1 = novo_st; return novo_st;}
```

```
short FirstOrder ( xn , koef1 ){
~~short~~ rez;
long
short novo_st, izlaz;

rez = Xn2+ koef1 · S2 · 2 + k · S1 · 2 ;

rez = rez + 0x8000;

~~rez~~ =

novo_st = rez >>16;

  S2 = Xn;
  S1 = novo_st;

  return novo_st; }

int main () {
  short s1, s2, s3, a0, a1, a2, a3 ... ;
  int i;
  long yn;
  for (i=0; i<Nsamp; i++){
    yn = First Order ( ~~ee~~ x[i], a0) + ---
        second Order ( x[i], a1[0], a1[1], b1[0], b1[1]
      + second Order( x[i], a2[0], a2[1], b2[0], b2[1]
    }
```