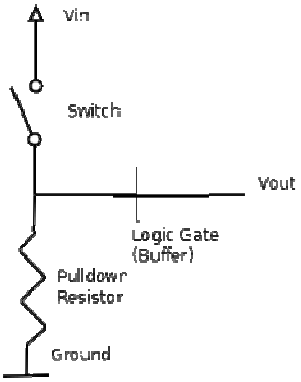
EA079 - Laboratório de Mico- e Mini-computadores: *Hardware*
Segunda Avaliação Individual - B

RA: _____ Nome:_____Turma: C

1. Na montagem de um circuito contendo diodos e capacitores eletrolíticos,
   a) (0.5 pt) como você identifica **visualmente** um diodo emissor de infra-vermelho?
   b) (0,5 pt) como você diferencia **visualmente** um capacitor eletrolítico de um cerâmico?

2. (1.0 pt) Determine a resistência do resitor *pull-down* no circuito para o pino Vout do microcontrolador MCF51CN128, considerando Vin=5V e a corrente máxima permitida I=0.5mA.



2. Em um projeto foram utilizados 3 componentes do microcontrolador MCF51CN128 para gerar/tratar interrupções: IRQ (PTC4), de prioridade 7; *output compare* (PTE4), de nível de prioridade 4; e *periodic interrupt*, de nível de prioridade 1.
   c) (1.0 pt) Qual(is) destes componentes geram interrupções por eventos externos? Justifique.
   d) (0.5 pt) Qual das interrupções é não-mascarável? Justifique.
   e) (0.5 pt) A qual dos três tipos de interrupção enquadra a seguinte descrição: "*When the value in the timer counter register matches the channel value register, an interrupt flag bit is set, and a selected output action is forced on the associated MCU pin. The output compare action is selected to force the pin to zero, force the pin to one, toggle the pin, or ignore the pin (used for software timing functions).*"
   f) (0,5 pt) Uma *periodic interrupt* **sempre suspende** o fluxo corrente de execução? Justifique.
   g) (0.5 pt) IRQ pode suspender o fluxo de tratamento da exceção *output capture*. Se quisermos que um fragmento de instruções dependentes do tratamento da exceção *output compare* seja executado sem interrupção, o que podemos fazer?

3. No projeto de um circuito de interface entre o microcontrolador MCF51CN128 e um par de emissor e receptor de infra-vermelho, foi utilizado 2 componentes do *Processor Expert* para captar os instantes de "chaveamento" do feixe de infra-vermelho (conduzindo/bloqueado), determinar o comprimento do objeto sobre uma esteira, e gerar um sinal de alerta quando o tamanho não esteja dentro das tolerâncias. A seguir são mostrados os trechos de código dos arquivos ProcessorExpert.c e Events.c, considerando

que a montagem do *hardware* seja tal que o sinal no pino PTE3 passe para nível lógico 0 quando o feixe for bloqueado e que o sinal do pino PTE4 passe para o nível lógico 1 quando a medida do objeto não esteja dentro das tolerâncias.

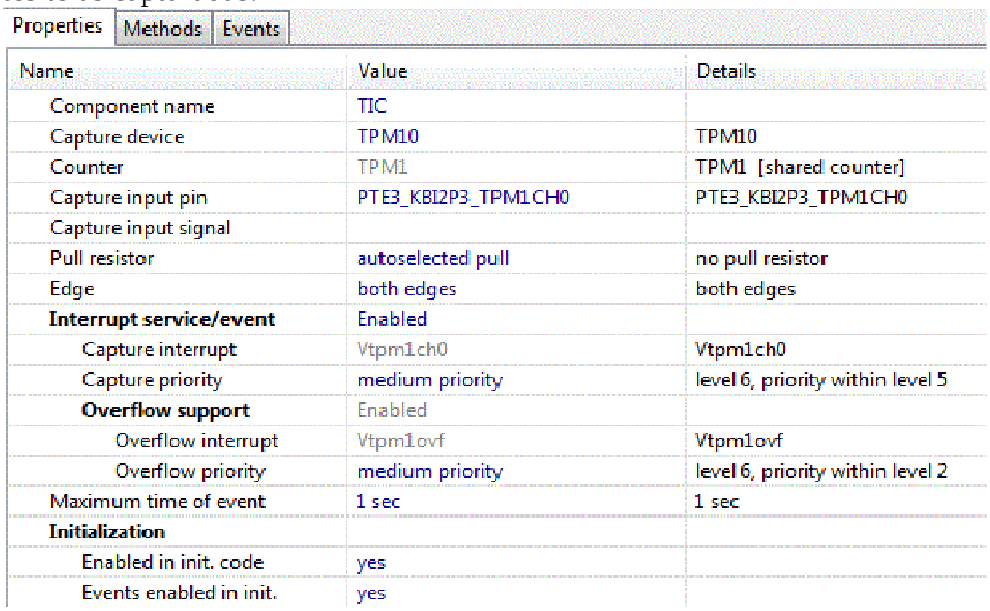
ProcessorExpert.c

```
#include "TIC.h"
/* Include shared modules, which
are used for whole project */

byte sensor_status;

void main(void)
{
  unsigned char c;
  PE_low_level_init();
  TOC_Disable();
  TOC_ClrValue();
  TIC_EnableEvent();

  sensor_status = 0;
  for(;;){}
}
```

Events.c

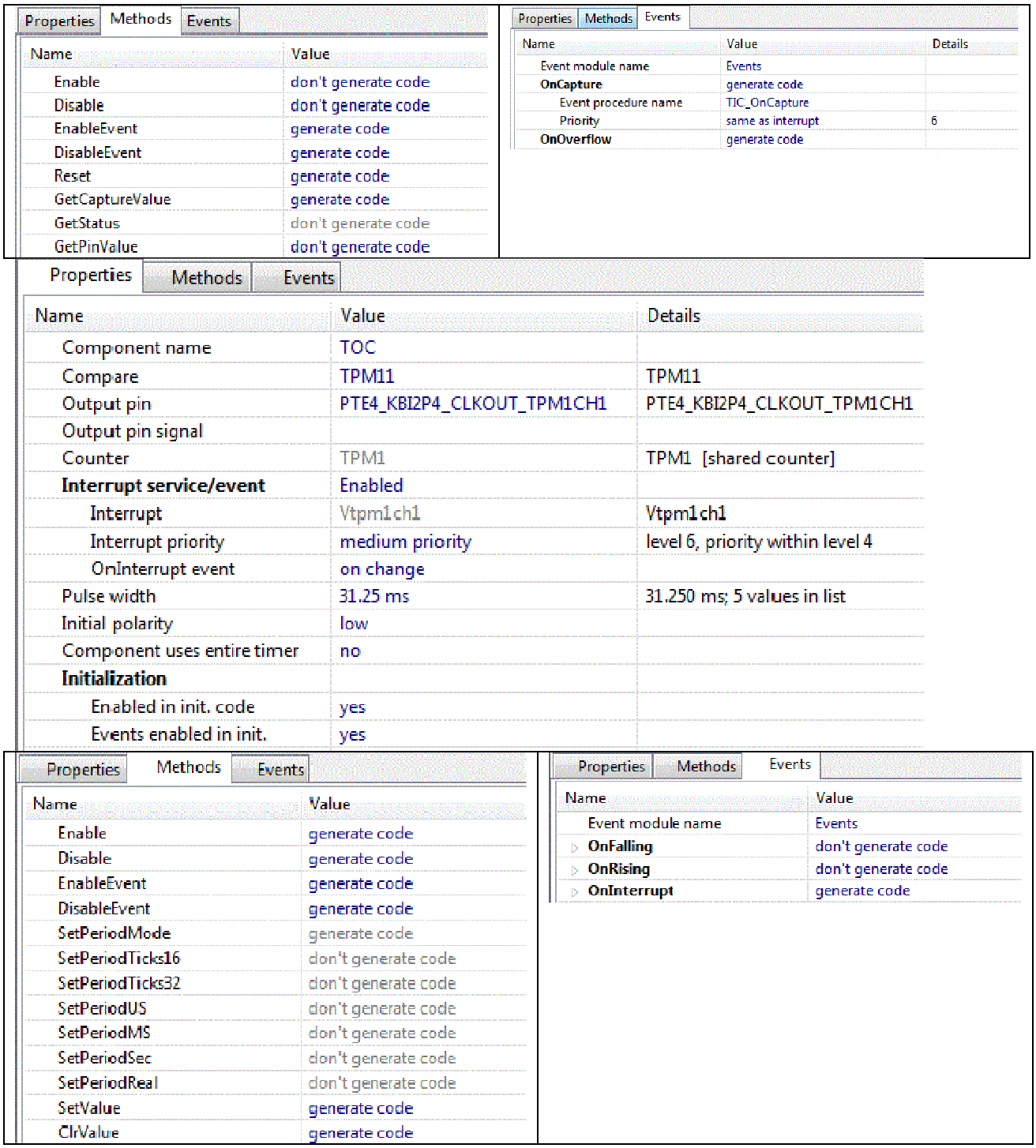
```
extern byte sensor_status;
word elapsed_time=0;
TIC_TCapturedValue inicio, final;
#define TOL ...
#define NOMINAL ...

void TOC_OnInterrupt(void)
{
 TOC_Disable();
 TOC_SetValue();
}
void TIC_OnCapture(void)
{
  if (sensor_status) {
    TIC_GetCaptureValue(&final);
    /* Determinar o comprimento
       do objeto: length */
    if (abs(length-NOMINAL)>TOL) {
       TOC_SetPeriodMode(1);
       TOC_Enable();
       TIC_DisableEvent();
    } else {
       TIC_Reset();
    }
    sensor_status = 0;
  } else {
    TIC_GetCaptureValue(&inicio);
    sensor_status=1;
  }
}
```

Os dois componentes foram configurados conforme os parâmetros apresentados nas seguintes telas capturadas:



Properties | Methods | Events

Name	Value	Details
Component name	TIC	
Capture device	TPM10	TPM10
Counter	TPM1	TPM1 [shared counter]
Capture input pin	PTE3_KBI2P3_TPM1CH0	PTE3_KBI2P3_TPM1CH0
Capture input signal		
Pull resistor	autoselected pull	no pull resistor
Edge	both edges	both edges
Interrupt service/event	Enabled	
Capture interrupt	Vtpm1ch0	Vtpm1ch0
Capture priority	medium priority	level 6, priority within level 5
Overflow support	Enabled	
Overflow interrupt	Vtpm1ovf	Vtpm1ovf
Overflow priority	medium priority	level 6, priority within level 2
Maximum time of event	1 sec	1 sec
Initialization		
Enabled in init. code	yes	
Events enabled in init.	yes	



Properties | Methods | Events

Name	Value
Enable	don't generate code
Disable	don't generate code
EnableEvent	generate code
DisableEvent	generate code
Reset	generate code
GetCaptureValue	generate code
GetStatus	don't generate code
GetPinValue	don't generate code

Properties | Methods | Events

Name	Value	Details
Event module name	Events	
OnCapture	generate code	
Event procedure name	TIC_OnCapture	
Priority	same as interrupt	6
OnOverflow	generate code	

Properties | Methods | Events

Name	Value	Details
Component name	TOC	
Compare	TPM11	TPM11
Output pin	PTE4_KBI2P4_CLKOUT_TPM1CH1	PTE4_KBI2P4_CLKOUT_TPM1CH1
Output pin signal		
Counter	TPM1	TPM1 [shared counter]
Interrupt service/event	Enabled	
Interrupt	Vtpm1ch1	Vtpm1ch1
Interrupt priority	medium priority	level 6, priority within level 4
OnInterrupt event	on change	
Pulse width	31.25 ms	31.250 ms; 5 values in list
Initial polarity	low	
Component uses entire timer	no	
Initialization		
Enabled in init. code	yes	
Events enabled in init.	yes	

Properties | Methods | Events

Name	Value
Enable	generate code
Disable	generate code
EnableEvent	generate code
DisableEvent	generate code
SetPeriodMode	generate code
SetPeriodTicks16	don't generate code
SetPeriodTicks32	don't generate code
SetPeriodUS	don't generate code
SetPeriodMS	don't generate code
SetPeriodSec	don't generate code
SetPeriodReal	don't generate code
SetValue	generate code
ClrValue	generate code

Properties | Methods | Events

Name	Value
Event module name	Events
OnFalling	don't generate code
OnRising	don't generate code
OnInterrupt	generate code

a) (2.0 pt) Em qual modo de operação foi configurado o canal 0 do módulo TPM1, se analisarmos a função gerada pelo *Processor Expert*? Justifique.

```
void TIC_Init(void)
{
  /* TPM1C0V:
BIT15=0,BIT14=0,BIT13=0,BIT12=0,BIT11=0,BIT10=0,BIT9=0,BIT8=0,BIT7=0,BI
T6=0,BIT5=0,BIT4=0,BIT3=0,BIT2=0,BIT1=0,BIT0=0 */
```

```
  setReg16(TPM1C0V, 0x00U);
  TIC_CntrState = 0x00U;               /* Clear variable */
  TIC_EnEvent = TRUE;                  /* Enable events */
  /* TPM1C0SC: CH0F=0,CH0IE=1,MS0B=0,MS0A=0,ELS0B=1,ELS0A=1,??=0,??=0
*/
  setReg8(TPM1C0SC, 0x4CU);
  /* TPM1SC: TOF=0 */
  clrReg8Bits(TPM1SC, 0x80U);          /* Reset overflow interrupt
request flag */
  /* TPM1SC: TOIE=1 */
  setReg8Bits(TPM1SC, 0x40U);          /* Enable overflow interrupt */
}
```

b) (2.0 pt) Explique a função da rotina **TIC_GetCaptureValue** conforme o código gerado pelo *Processor Expert*.

```
#define TIC_Reset() \
     (TIC_CntrState = TPM1CNT , (byte)ERR_OK)

#define TIC_GetCaptureValue(Value) \
  /*lint -save  -e926 -e927 -e928 -e929 Disable MISRA rule (11.4)
checking. */\
  (((*(TIC_TCapturedValue*)(Value) = TPM1C0V), \
  (*(TIC_TCapturedValue*)(Value) -= TIC_CntrState)), \
  ERR_OK) \
     /*lint -restore Enable MISRA rule (11.4) checking. */
```

c) (1.0 pt) Explique o procedimento de controle do tamanho das peças com uso dos dois componentes.