PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ
ESCOLA POLITÉCNICA
ENGENHARIA DE COMPUTAÇÃO
MICROPROCESSADORES

PROJETO PARDAL
DOCUMENTAÇÃO

CURITIBA, 2012
PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ
ESCOLA POLITÉCNICA
ENGENHARIA DE COMPUTAÇÃO
MICROPROCESSADORES

ANDRÉ COLI N
LUCAS A. JOHNSON
LUCAS GUIMARÃES GONÇALVES

PROJETO PARDAL
DOCUMENTAÇÃO

Documentação do projeto integrado realizada pelos alunos André Colin, Lucas Guimarães e Lucas Johnson para a disciplina de Microprocessadores do curso de Engenharia de Computação.

CURITIBA, 2012

1. RESUMO

Essa documentação descreve as informações, códigos e desenhos esquemáticos
relacionados ao projeto integrado Pardal.

A ideia do projeto foi dada pelo aluno Lucas A. Johnson e realizada também pelo aluno Lucas Guimarães Gonçalves e André Colin para a disciplina de Microprocessadores, sob orientação do professor Afonso Ferreira Miguel.

O Projeto Pardal tem como finalidade detectar infrações de trânsito. Tendo acoplado ao seu conjunto, um sistema que fotografa o veículo infrator. Em muitos radares, a tecnologia utilizada é a de sinais sonoros para detecção de eco, sendo que os mesmos são do tipo radar móvel, e ainda radares que utilizam bobinas eletromagnéticas, que são do tipo radar fixo. Esse projeto tem como base sensores infravermelho que detectam o ponto de início e o ponto final de um corpo, sendo assim, calculando sua velocidade média em milímetros por segundo (mm/s) . A criação de um sinaleiro para criar uma situação de avanço do sinal e de parada sobre a faixa de pedestres, também faz parte do desenvolvimento do projeto. O detector de imagem constituirá de uma câmera digital que através de um sinal do micro controlador PIC, irá capturar ou não a imagem do veículo em movimento.

2. OBJETIVOS

Nesse item será enfocado o objetivo do projeto com o usuário e o objetivo para realização do mesmo.

2.1. DO PROJETO

Atualmente, em ambientes urbanos, a necessidade da criação de dispositivos de fiscalização se torna cada vez maior, sejam eles para controlar a velocidade de um veículo, seja para controlar o avanço no sinal vermelho e parada encima da faixa de pedestres.

Logo, o PARDAL entra como uma alternativa, tornando possível a implementação de um dispositivo de fiscalização no trânsito.

2.2. ESPECÍFICO PARA A REALIZAÇÃO DO PROJETO

- Relatar o projeto e criar um cronograma para o mesmo.
- Montagem da maquete.
- Programação dos componentes digitais.
- Montagem das placas eletrônicas.
- Implementação da placa no piso.
- Implementação da interface de alimentação e ativação.

3. LISTA DE COMPONENTES

Nesse item serão mostrados os materiais utilizados para a construção do projeto.

3.1. DESCRIÇÃO

<Quantidade> <Nome>: <Descrição>
1 PIC12F675 (SIP 8 pinos): Microcontrolador utilizado para fazer a conversão A/D do sinal dos sensores IR e ativar o dispositivo externo quando o veículo cometer a infração.
3 Resistores (5,6k; 1k; 100): Para uso específico.
2 emissores e 2 sensores IR: Utilizados para captar e emitir o sinal IR.
1 Placa de Fenolite Cobreada: Placa para montagem do circuito.
Fios: Utilizados para fazer conexoes externas.
Isopor e cartolina, para montar a maquete.
Tubo de pvc e leds para construir o sinaleiro.
1 Carrinho de controle remoto.

3.2 – IMAGENS DOS COMPONENTES ELETRÔNICOS

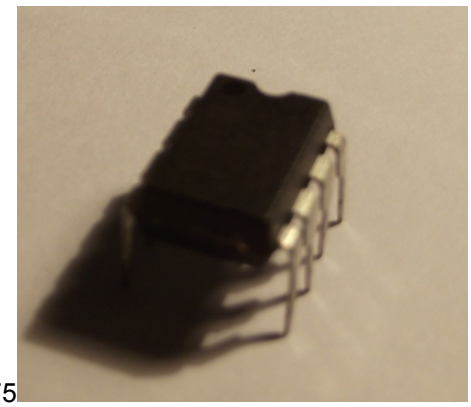

Figura 1: Microcontrolador PIC12F675

4. ETAPAS DETALHADAS DE REALIZAÇÃO DO PROJETO

O plano de trabalho foi utilizado como um guia para a montagem de um cronograma, que foi cumprido pelos integrantes da equipe para a construção do projeto.

Para montagem da maquete foram utilizados um tubo de pvc para construir o sinaleiro, isopor e cartolina para o desenho da rua e calçada..

O velocidade máxima permitida pelo radar é 1km/h e o tempo para a foto ser tirada caso o veículo fique encima da faixa é de ser 5 segundos. Quando o carro interromper o primeiro sensor, um *timer* (temporizador) interno do pic é ativado e desativado quando o segundo sensor é interrompido, o tempo é verificado e o carro é fotografado quando o tempo do *timer* não exceder um limite. A parada na faixa é verificada quando o carro sair do segundo sensor, caso de mais de 5 segundos o carro é fotografado. Todas as fotos são tiradas quando o carro passar pelo segundo sensor.

Para programar o PIC, foi necessário um compilador Assembly e uma ferramenta de programação PICStart Plus. No PIC, foram utilizados um pino de entrada analógica para receber a tensão de saída do Sensor de Hall e um outro pino de saída digital para ativar o amplificador. O encarregado de realizar essa tarefa foi Lucas, que também forneceu as ferramentas necessárias.

5. CÓDIGO FONTE PIC

```
        list p=12f675
        #include <p12f675.inc>

        __CONFIG _INTRC_OSC_NOCLKOUT & _WDT_OFF & _PWRTE_ON & _MCLRE_OFF & _CP_OFF
& _CPD_OFF

; Variáveis (0x20 = Início da RAM de acesso livre)
        bytecontrole equ 0x20      ; bit 0: Específico, bit 1: Sinal vermelho, bit 2: Parada na faixa, bit 3: Específico.
        contador equ 0x21

        contador2 equ 0x22

;  ------------------------------------------------------------------------------------
        org 0x000 ; RESET.
        goto inicio
; ------------------------------------------------------------------------------------
        org 0x004 ; INTERRUPÇÃO.
        retfie ; Retorna.
; ------------------------------------------------------------------------------------

inicio:
        call configuraPrograma

codigo:
```

```
        btfsc PIR1, 0           ; Verifica overflow do timer.
        call incrementaContador
        bcf PIR1, 0

        bsf ADCON0, 0

        bcf ADCON0, 1           ; Prepara para pegar a amostragem (A/D).
        bsf ADCON0, 1           ; Pega amostragem (A/D).

        call delayAD            ; Espera AD

        bcf ADCON0, 0

        movlw 0x10
        bsf STATUS, C
        subwf ADRESH, 0

        btfss STATUS, C
        goto carroPassou

        btfsc bytecontrole, 0
        call mudaSensor

        goto codigo

mudaSensor:
        movlw 0x20
        bsf STATUS, C
        subwf contador2, 0

        btfsc STATUS, C
        goto naoincrementa

        incf contador2, 1

        goto codigo

naoincrementa:
        bcf T1CON, TMR1ON       ; Desativa Timer 1.

        call testeFaixa

        bcf GPIO, 2

        btfsc GPIO, 3
        call cameraFoto

        bcf ADCON0, 2           ; Muda para primeiro sensor.

        clrf contador2

        bcf bytecontrole, 2
        clrf contador

        bcf bytecontrole, 0
```

```
        goto codigo

carroPassou:
        btfsc bytecontrole, 0
        goto codigo

        btfss ADCON0, 2
        goto aTimer

        btfsc ADCON0, 2
        goto dTimer

aTimer:
        call ativaTimer
        bsf ADCON0, 2           ; Muda para segundo sensor.
        goto codigo

dTimer:
        call desativaTimer
        goto codigo

; ------------------------------------------------------------------------------

; FUNÇÕES

configuraPrograma:
        movlw B'00001011'       ; Move 00000011 (binário) para o registrador W.
        bsf STATUS, RP0         ; Muda para o banco 1 (RP0 = 1).
        movwf TRISIO            ; Bota os pinos GP0 e GP1 (pino 6 e 7 do diagrama) como entrada e o resto como
saída.
        movlw B'00000011'       ; Move 00000011 (binário) para o registrador W.
        movwf ANSEL             ; Seta entradas analógicas AN0 e AN1, o resto como entrada/saída digital.
        bcf STATUS, RP0         ; Muda para banco 0.

        bcf GPIO, 2             ; Apaga LED 1.
        bcf GPIO, 4             ; Não ativa câmera.

        bcf T1CON, TMR1ON       ; Desativa Timer 1.
        movlw b'00110000'       ; Configurando Timer 1. (Clock/4)/PreScale = (Clock/4)/8 = Clock/32.
                        ; P/ um clock de 4000000, Frequência do Timer1 = 125000 Hz =~ 0,5s p/ overflow.
        movwf T1CON
        bcf PIR1, 0             ; Bit de interrupção do Timer 1.
        clrf contador           ; Contador de interrupções do Timer 1.

        bcf ADCON0, 3
        bcf ADCON0, 2

        clrf contador2

        bcf bytecontrole, 0     ; Específico. -
;       bcf bytecontrole, 1     ; Sinal vermelho.
        bcf bytecontrole, 2     ; Faixa. -
        ;bcf bytecontrole, 3    ; Específico. -

        return
```

```
; end configuraPrograma

incrementaContador:
        bsf bytecontrole, 2

        movlw 0xFD
        bsf STATUS, C

        subwf contador, 0

        btfss STATUS, C
        incf contador, 1

        return
; end incrementaContador

delayAD:
        nop
        nop
        nop
        nop
        nop
        nop

        return
; end delayAD

ativaTimer:
        bcf GPIO, 4
        bsf GPIO, 2

        clrf TMR1L
        clrf TMR1H              ; Elimina valor presente no Timer 1.

        bsf T1CON, TMR1ON       ; Inicia Timer 1.

        return
; end ativaTimer

desativaTimer:
        btfsc bytecontrole, 2
        goto dTimerFinal

        bcf T1CON, TMR1ON       ; Desativa Timer 1.

        ; Teste de velocidade.
        call ativarTesteDeTempo

        bsf T1CON, TMR1ON       ; Ativa Timer 1.

dTimerFinal:
        bsf bytecontrole, 0

        return
; end desativaTimer
```

```
ativarTesteDeTempo:
        movlw 0x23              ; 1 km/h
        bsf STATUS, C
        subwf TMR1H, 0

        btfss STATUS, C
        call cameraFoto

        return
; end ativarTesteDeTempo

testeFaixa:
        movlw 0x0A
        bsf STATUS, C
        subwf contador, 0

        btfsc STATUS, C
        call cameraFoto

        return
; end testeFaixa

cameraFoto:
        bsf GPIO, 4

        return
; end cameraFoto




controlaSinaleiro:

        return
; end controlaSinaleiro

        END
```

6. CONCLUSÃO

O projeto integrado foi um grande projeto de aprendizado, desde o entendimento de como funciona um sensor infravermelho ate a programção de um microcontrolador, incluindo a montagem das placas. Além disso, a equipe ganha como experiencia também um avanço na organização de ideias e tomada de decisões frente a problemas encontrados.

A equipe agradece ao professor Afonso Ferreira Miguel e a PUCPR, que nos disponibilizou os laboratórios e as ferramentas.