UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO – BACHARELADO

PROTÓTIPO DE UM COMPUTADOR DE BORDO PARA AUTOMÓVEIS BASEADO NA ARQUITETURA ARM

RAFAEL DE SOUZA

BLUMENAU
2006

2006/2-23

RAFAEL DE SOUZA

PROTÓTIPO DE UM COMPUTADOR DE BORDO PARA AUTOMÓVEIS BASEADO NA ARQUITETURA ARM

Trabalho de Conclusão de Curso submetido à Universidade Regional de Blumenau para a obtenção dos créditos na disciplina Trabalho de Conclusão de Curso II do curso de Ciências da Computação — Bacharelado.

Prof. Miguel Alexandre Wisintainer - Orientador

BLUMENAU
2006

2006/2-23

PROTÓTIPO DE UM COMPUTADOR DE BORDO PARA AUTOMÓVEIS BASEADO NA ARQUITETURA ARM

Por

RAFAEL DE SOUZA

Trabalho aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II, pela banca examinadora formada por:

Presidente:	Prof. Miguel Alexandre Wisintainer – Orientador, FURB
Membro:	Prof. Antonio Carlos Tavares – FURB
Membro:	Prof. Mauro Marcelo Mattos – FURB

Blumenau, 13 de dezembro de 2006

Dedico este trabalho a todos os amigos, especialmente aqueles que me ajudaram diretamente na realização deste.

AGRADECIMENTOS

À Deus, pela sabedoria, inteligência e força, fazendo-me chegar até aqui sem desistir, guiando e me acompanhando todos os dias.

Aos meus pais, pela formação e educação que recebi. Obrigado pela compreensão, confiança e força que sempre recebi de vocês.

Aos meus amigos, pelos empurrões e cobranças.

Ao meu orientador, professor Miguel Alexandre Wisintainer, pelos ensinamentos e apoio na realização deste trabalho.

“Algo só é impossível até que alguém duvide e acabe provando o contrário.”

Albert Einsten

RESUMO

Este trabalho apresenta um protótipo de computador de bordo para automóveis utilizando o microontrolador ARM da Philips, representa funções como velocidade veicular, distância percorrida, rotações por minuto do motor, temperatura do sistema de arrefecimento do motor, nível de combustível, e carga da bateria, apresentando os dados em um *display* de cristal líquido. O desenvolvimento deste sistema proporcionou a utilização do compilador C do ambiente de desenvolvimento WinArm.

Palavras-chave: Computador de bordo. Velocímetro digital. Microcontrolador ARM. LCD.

ABSTRACT

This work presents an automobile computer prototype using Philips ARM microcontroller, represents functions as vehicle speed, covered distance, engine rotations per minute, cooling system temperature of the engine, fuel level, and battery load, showing the data in a liquid crystal display. The system development provided to the use of compiler C using the environment of WinArm development.

Key-words: Automobile computer. Digital speedometer. ARM microcontroller. LCD.

LISTA DE ILUSTRAÇÕES

Figura 1 – Pinagem microcontrolador ARM LPC2294 15
Figura 2 – Diagrama de blocos do microcontrolador ARM LPC2294 16
Figura 3 – Utilização de registradores 17
Figura 4 – Leitura da porta analógica AIN0 17
Figura 5 – Interrupção do *timer* 0 18
Figura 6 – Interrupção EINT1 18
Figura 7 – Interrupção sem vetor 19
Figura 8 - Sensor de temperatura tipo 1 e tipo 2 20
Figura 9 - Sensor de velocidade da Ford 20
Figura 10 – *Display* de cristal líquido WH1602L 22
Figura 11 – Diagrama de casos de uso do sistema 25
Figura 12 – Diagrama de atividades da ferramenta 26
Figura 13 – Diagrama de atividades da configuração 28
Figura 14 – Ambiente de desenvolvimento do WinArm 30
Figura 15 – LPC2000 *Flash Utility* 31
Figura 16 – Ambiente de simulação do Proteus 31
Figura 17 – Cálculo da velocidade veicular 32
Figura 18 – Cálculo do percentual de combustível no tanque do veículo 33
Figura 19 – Executando o Hyper Terminal 33
Figura 20 – Configuração do Hyper Terminal 34
Figura 21 – Menu principal de configurações 34
Figura 22 – Menu velocidade 35
Figura 23 – Configuração gravada na memória EEPROM 36
Figura 24 – Menu temperatura 36
Figura 25 – Menu combustível 37
Figura 26 – Menu RPM 38
Figura 27 – Tela de saudação 38
Figura 28 – Informação de menu 1 39
Figura 29 – Dados do menu 1 39
Figura 30 – Informação de menu 2 39
Figura 31 – Dados do menu 2 40

Figura 32 – Informação de menu 3.....40
Figura 33 – Dados do menu 3.....40
Figura 34 – Informação de menu 4.....41
Figura 35 – Dados do menu 4.....41
Figura 36 – Hardware do protótipo 42

LISTA DE QUADROS

Quadro 1 – Tipos de sensores.....21
Quadro 2 – Pinagem display WH1602.....22
Quadro 3 – Descrição das atividades.....26
Quadro 4 – Descrição das atividades de configuração.....29

LISTA DE SIGLAS

ADC – *Analog to Digital Converter*

AIN – *Analog INput*

ARM – *Advanced Risc Machines*

CAN – *Controller Area Network*

EEPROM – *Electrically Erasable Programable Read Only Memory*

EINT – *External INTerrupt*

GPS – *Global Position System*

IOCLR – *Input Output CLEar*

IODIR – *Input Output DIRection*

IOPIN – *Input Output PIN*

IOSET – *Input Output SET*

LCD – *Liquid Crystal Display*

MP3 – *Moving Picture expert group audio layer 3*

PC – *Personal Computer*

PINSEL – *PIN function SELect*

PWM – *Pulse Width Modulator*

RAM – *Random Access Memory*

RPM – Rotações Por Minuto

TCC – Trabalho de Conclusão de Curso

UML – *Unified Modeling Language*

SUMÁRIO

1 INTRODUÇÃO 12
1.1 OBJETIVOS DO TRABALHO 12
1.2 ESTRUTURA DO TRABALHO 13
2 FUNDAMENTAÇÃO TEÓRICA 14
2.1 MICROCONTROLADOR ARM 14
2.2 COMPUTADOR DE BORDO 19
2.2.1 Sensores automotivos 19
2.3 *DISPLAY* LCD 21
2.4 TRABALHOS CORRELATOS 22
3 DESENVOLVIMENTO DO SISTEMA 24
3.1 REQUISITOS PRINCIPAIS 24
3.2 ESPECIFICAÇÃO 24
3.2.1 Especificação do protótipo 25
3.2.1.1 Diagrama de casos de uso 25
3.2.1.2 Diagrama de atividades 26
3.2.1.2.1 Atividade de configuração 28
3.3 IMPLEMENTAÇÃO 29
3.3.1 Técnicas e ferramentas utilizadas 30
3.3.2 Operacionalidade da implementação 33
3.4 RESULTADOS E DISCUSSÃO 43
4 CONCLUSÕES 44
4.1 EXTENSÕES 44
REFERÊNCIAS BIBLIOGRÁFICAS 45

1 INTRODUÇÃO

Computadores de bordo e *check control* são itens que há décadas vem sendo incorporados aos veículos europeus. Já no Brasil, um carro possuir computador de bordo é pouco usual no trânsito. Veículos populares possuem apenas algumas funções básicas como hodômetro digital e velocímetro analógico. Já os veículos mais sofisticados possuem diversos recursos incorporados como carga da bateria, consumo médio, consumo instantâneo, quantidade de litros de combustível consumidos, distância que pode ser percorrida com a quantidade de combustível restante, alerta sonoro ou visual para quando ultrapassar a velocidade estipulada ou limite de rotações por minuto (RPM) do motor.

Tanto o computador de bordo quanto o *check control* funcionam por meio de sensores eletrônicos. O *check control* acusa através de imagens e luzes se há problema com sistema elétrico, se as portas estão abertas, se o cinto de segurança não está conectado, entre outros (CHECK..., 1999).

Os constantes avanços tecnológicos são impulsionadores dos novos conceitos em controle de tráfego viário. Veículos equipados com computadores de bordo poderão receber instruções do controle do trânsito central sobre a melhor rota para se alcançar um determinado lugar de destino. O computador de bordo também poderá informar o tempo de duração da viagem através da velocidade atual do veículo (ASSOCIAÇÃO BRASILEIRA DE MONITORAMENTO E CONTROLE ELETRÔNICO DE TRÂNSITO, 2003).

Baseado nos conceitos apresentados, foi desenvolvido um computador de bordo utilizando microcontrolador que, através dos sensores presentes em um veículo, coleta informações essenciais, transformando os dados obtidos em informações consistentes, além de realizar projeções através dos dados coletados anteriormente. O computador de bordo pode ser ligado a um computador pessoal (PC) através da porta serial para configuração do sistema conforme as características do veículo a ser utilizado. Os dados processados no microcontrolador são exibidos em um *display* de cristal líquido (LCD).

1.1 OBJETIVOS DO TRABALHO

O objetivo deste trabalho é desenvolver software e hardware microcontrolado que

interajam com os sensores de velocidade, temperatura de água e combustível presentes em um veículo.

Os objetivos específicos do trabalho são:

a) construir a interface entre os diversos sensores presentes no veículo utilizando o microcontrolador ARM LPC2294;
b) permitir controle de velocidade, emissão de alertas e registro de excessos de velocidade e RPM;
c) disponibilizar quatro hodômetros parciais contendo velocidade média, velocidade máxima, consumo médio, máxima velocidade angular do motor alcançada no período de um minuto (RPM) e tempo decorrido, ficando os dados armazenados até o reinicio do hodômetro;
d) usar porta serial para configuração do sistema, informando dados dos sensores a serem utilizados;
e) mostrar os dados através de um LCD.

1.2 ESTRUTURA DO TRABALHO

Esse trabalho está estruturado como descrito a seguir:

a) no capítulo 2, serão apresentadas todas as áreas relacionadas com a fundamentação teórica, necessárias para o desenvolvimento deste trabalho, sendo elas: Microcontrolador ARM, Computador de bordo, sensores automotivos e *Display* LCD;
b) no capítulo 3 é apresentado todo o processo de desenvolvimento do software, incluindo os requisitos, algoritmos, técnicas, trechos de código, etc, utilizados durante esta etapa;
c) no capítulo 4 serão apresentadas as conclusões, análise dos resultados obtidos e sugestões para extensão do presente trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

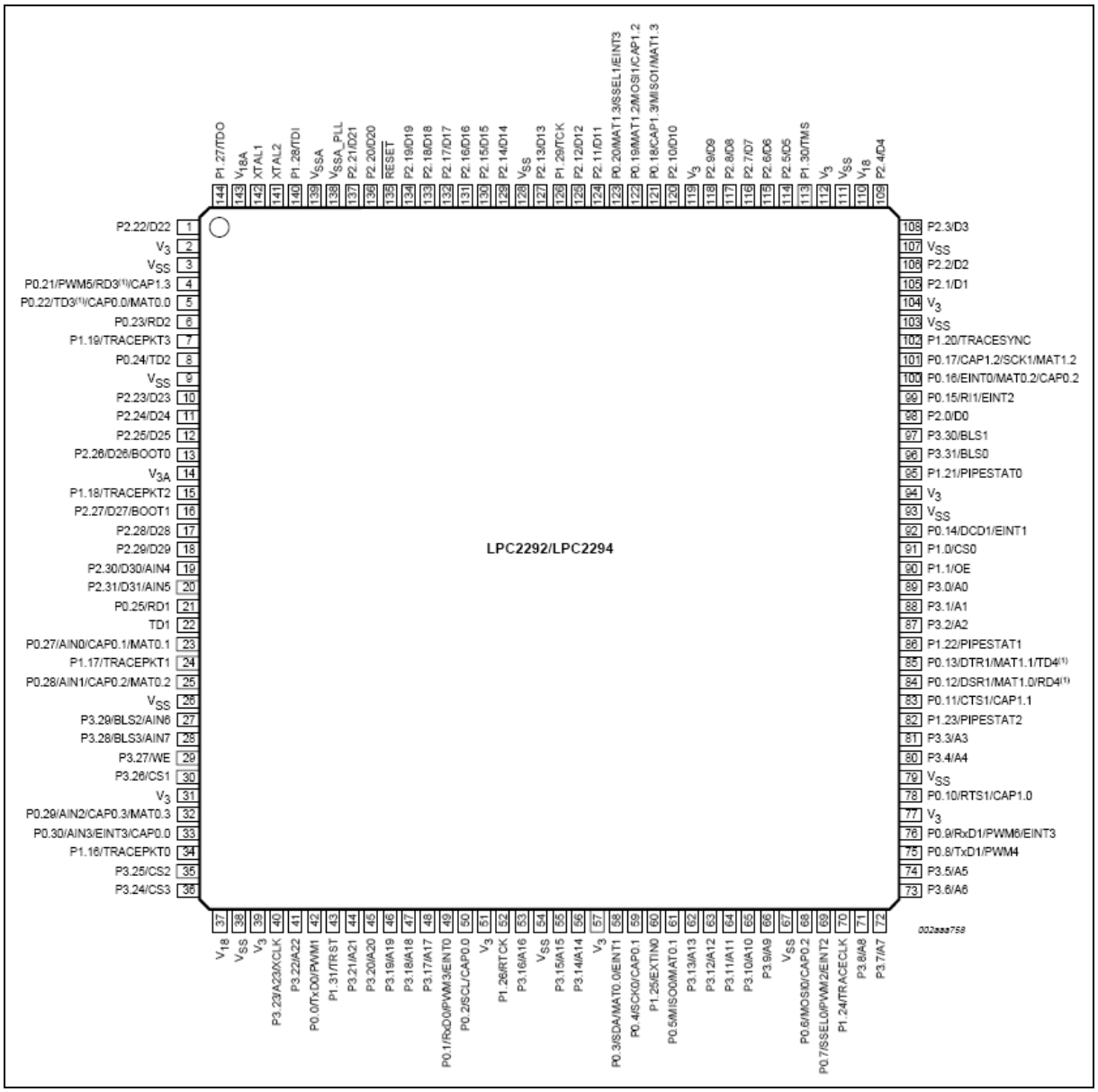
Esse capítulo está organizado em quatro seções. A primeira apresenta as principais características do microcontrolador ARM. A segunda seção apresenta algumas informações sobre computadores de bordo. A terceira seção apresenta informações sobre *display* LCD. Por fim a quarta seção apresenta os trabalhos correlatos.

2.1 MICROCONTROLADOR ARM

Um microcontrolador é um computador programável em um único *chip* de forma otimizada e compacta para controlar dispositivos eletrônicos. É uma espécie de microprocessador, com memória e interfaces de entrada e saída integrados, o mesmo tipo usado em PCs. Os microcontroladores são componentes utilizados em muitos tipos de equipamentos eletrônicos, podendo ser encontrados em forno de microondas, máquinas de lavar, telefones, entre outros (MICROCONTROLADOR, 2006).

Os microcontroladores ARM da família LPC22XX são baseados em um processador central de 16/32 bits, sendo que o microcontrolador possui 256 kbytes de memória interna que pode ser gravada externamente ou pela própria aplicação e 16 kbytes de memória de acesso randômico (RAM). Pode operar na velocidade de até 60 MHz com baixo consumo de energia. Para aplicações com espaço crítico em memória, é possível executar instruções de 16 bits denominadas *thumb*, que podem ter o seu código reduzido em 30 por cento do tamanho se comparado às instruções normais de 32 bits, porém com uma perda de desempenho de aproximadamente 40 por cento. É constituído de 144 pinos, sendo 122 pinos de entrada e saída, diversos *timers* de 32 bits, 8 canais de conversão analógicos para digital (ADC) de 10 bits, 4 interfaces para controle de rede (CAN), canal de modulação por largura de pulso (PWM) e até 9 pinos externos de interrupção. Possui 16 vetores de interrupção para uso de interrupções simultâneas. A tensão dos pinos de entrada e saída deverá ser entre 0 e 3,3 volts, tanto para portas digitais como analógicas. Estes microcontroladores são particularmente apropriados para aplicações automotivas e industriais (PHILIPS ELETRONICS, 2006).

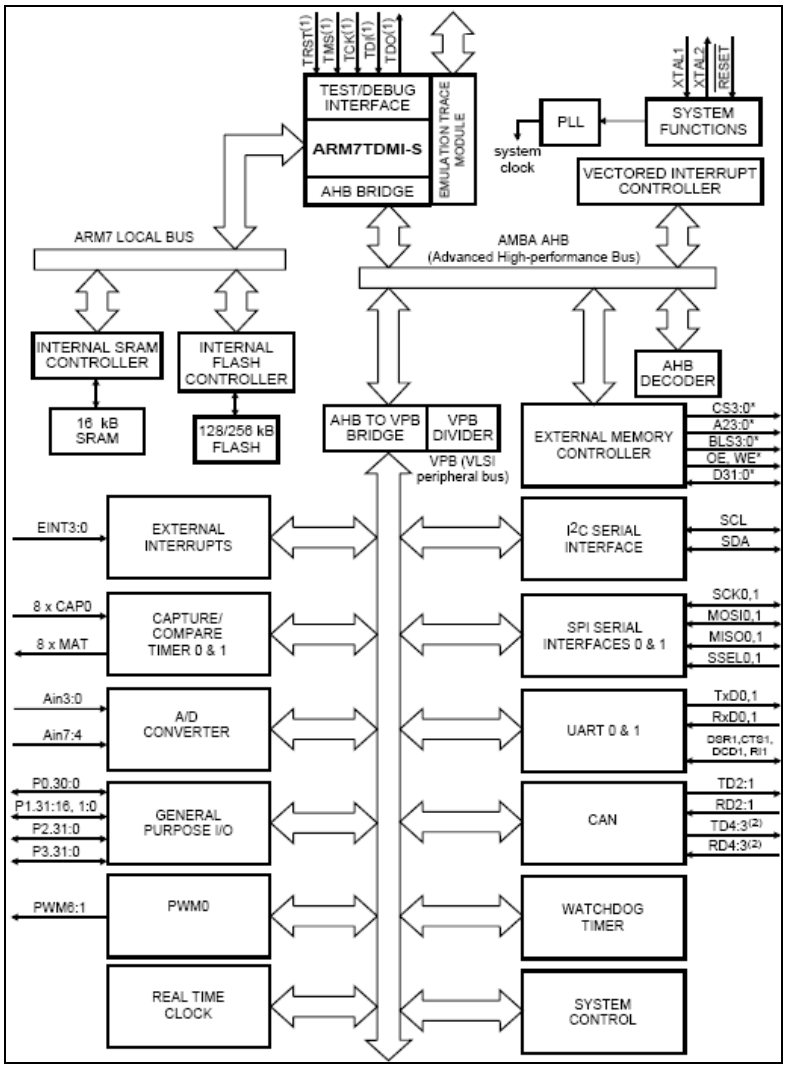
A Figura 1 mostra a pinagem do microcontrolador ARM LPC2294.



Fonte: Future Eletrônics (2006).

Figura 1 – Pinagem microcontrolador ARM LPC2294

A Figura 2 demonstra o diagrama de blocos do microcontrolador ARM LPC2294.



Fonte: Future Eletrônics (2006).

Figura 2 – Diagrama de blocos do microcontrolador ARM LPC2294

Os pinos do microcontrolador estão divididos em 4 grupos, sendo elas P0, P1, P2 e P3, cada grupo possui 31 pinos distintos, por exemplo para acessar o pino 3 do grupo P0 se usaria P0.3.

O microcontrolador apesar de possuir diversos pinos de entrada e saída ainda pode possuir mais de uma função específica para cada um deles utilizando o registrador PINSEL para selecionar, lembrando que como existe 4 grupos de pinos então temos PINSEL0 a PINSEL3.

Para definir se determinados pinos serão utilizados como entrada ou saída, deverão ser utilizados os registradores IODIR0 a IODIR3, definindo o bit como 0 para entrada e 1 para saída. Para ativar um pino, utilizar os registradores IOSET0 a IOSET3 e IOCLR0 a IOCLR3 para apagar. Para verificação do nível lógico, utilizar IOPIN0 a IOPIN3 (PHILIPS ELETRONICS, 2006).

A Figura 3 demonstra a utilização dos registradores IODIR, IOSET, IOCLR e IOPIN.

```
#define LED1    10              // Define o pino 10 para o LED
#define BOTAO1  11              // Define o pino 11 para o botão

void main(void)
{
  IODIR0 |= (1<<LED1);          // Define o pino P0.10 como saída(1) para o LED1,
                                // pegando o valor 1 e dando um shift para a esquerda
                                // de 10 posições, resultando no valor binario 1000000000
                                // e comparando com operador lógico "OU"
                                // atribui ao IODIR0 com comando OR, setando apenas o bit da
                                // posição 10.
  IODIR0 &= ~(1<<BOTAO1);       // Define o pino P0.11 como entrada(0) para o BOTAO1,
                                // pegando o valor 1 e dando um shift para a esquerda
                                // de 11 posições e negando o resultado,
                                // resultando no valor binário 01111111111 e comparando
                                // com o operador lógico "E", zerando apenas o bit da
                                // posição 11.

  while (1);                    // Deixa o programa em loop
  {
    if (IOPIN0 & (1<<BOTAO1))   // Verifica se o botão está pressionado
      IOSET0 |= (1<<LED1);      // Acende o LED
    else
      IOCLR0 |= (1<<LED1);      // Apaga o LED
  }
}
```

Figura 3 – Utilização de registradores

O microcontrolador ARM possui 8 portas analógicos de 10 bits de dados cada uma, resultando em um valor digital de 0 a 1023, são elas:

a) AIN0 pino P0.27;

b) AIN1 pino P0.28;

c) AIN2 pino P0.29;

d) AIN3 pino P0.30;

e) AIN4 pino P2.30;

f) AIN5 pino P2.31;

g) AIN6 pino P3.29;

h) AIN7 pino P3.28.

A Figura 4 apresenta uma rotina de leitura analógica na porta AIN0.

```
int main(void)
{
  unsigned int val;

  ADCR  = 0x00270601;               // Seta parâmetros do A/D e seleciona o canal AIN0
  ADCR |= 0x01000000;               // Inicia conversão A/D
  do
  {
    val = ADDR;                     // Lê os dados do registrador do A/D
  } while ((val & 0x80000000) == 0);// Fica em loop até terminar a conversão e 'val' ter o valor do A/D
  val = ((val >> 6) & 0x03FF);      // Faz um shift de 6 bits para a direita e pega os 10 primeiros bits
                                    // que é o valor do A/D (0 a 1023)
  while (1);
}
```

Fonte: adaptado de Trevor (2006, p. 48).

Figura 4 – Leitura da porta analógica AIN0

O microcontrolador ARM possibilita a criação de interrupção de *timers* que serão executados em períodos de tempo predefinidos. Sempre que ocorrer alguma interrupção, a

execução do sistema será desviada para atender a interrupção, após a execução da mesma é retornado para o local que estava sendo executado anteriormente.

A Figura 5 apresenta a criação de uma rotina de interrupção do *timer* 0.

```
int main(void)
{
  T0MR0 = 0x00000010;                 // Seta o ciclo timer
  T0MCR = 0x00000003;                 // Gera interrupção e reseta o contador
  T0TCR = 0x00000001;                 // Habilita o timer
  VICVectAddr4   = (unsigned)T0isr;   // Seta o vetor de interrupção 4 para a rotina T0isr
  VICVectCntl4   = 0x00000024;        // Seta o canal;
  VICIntEnable |= 0x00000010;         // Habilita a interrupção

  while(1);
}

void T0isr(void) __irq
{
  // Toda vez que ocorrer a interrupção do timer irá chamar esta rotina
  T0IR        = 0x00000001;           // Limpa o flag de interrupção
  VICVectAddr = 0x00000000;           // Escreve sinal para final da interrupção
}
```

Fonte: adaptado de Trevor (2006, p. 80).

Figura 5 – Interrupção do *timer* 0

O microcontrolador ARM possui 4 portas de interrupções externas divididas em 9 pinos, que serão executados sempre que ocorrer um pulso no pino de interrupção, são elas:

a) EINT0 pino P0.1 ou P0.16;

b) EINT1 pino P0.3 ou P0.14;

c) EINT2 pino P0.7 ou P0.15;

d) EINT3 pino P0.9 ou P0.20 ou P0.30;

A Figura 6 apresenta a criação de uma rotina de interrupção externa, no qual o sistema irá desviar a sua execução sempre que ocorrer um pulso na porta EINT1.

```
int main(void)
{
  PINSEL0      = 0x20000000;          // Habilita a interrupção EXTINT1
  VICVectAddr  = (unsigned)EXTINT1;   // Seta o vetor de interrupção 0 para a rotina EXTINT
  VICVectCntl0 = 0x0000002F;          // Seta o canal da EXTINT1 no VIC
  VICIntEnable = 0x00008000;          // Habilita a interrupção

  while (1);
}

void EXTINT1(void) __irq
{
  EXTINT      = 0x00000002;           // Limpa o flag de interrupção
  VICVectAddr = 0x00000000;           // Escreve sinal para final da interrupção
}
```

Fonte: adaptado de Trevor (2006, p. 71).

Figura 6 – Interrupção EINT1

O microcontrolador ARM permite a utilização de até 16 interrupções simultâneas, setando um vetor para cada uma delas, porém é possível utilizar um vetor de interrupção padrão, denominado de interrupção sem vetor, o qual será executado sempre quando houver qualquer tipo de interrupção, sendo necessário verificar manualmente o tipo de interrupção

gerada. A Figura 7 apresenta a criação de uma interrupção sem vetor.

```
int main(void)
{
  PINSEL0 = 0x20000000;                       // Habilita a interrupção EXTINT0
  VICDefVectAddr = (unsigned)NonVectoredIRQ; // Seta o vetor de interrupção
  VICIntEnable   = 0x8000;                    // Habilita a interrupção

  while(1);
}

void NonVectoredIRQ(void) __irq
{
  if (VICIRQStatus & 0x00008000)              // Testa se interrupção EXTINT0
  {
    EXTINT = 0x00000002;                      // Limpa o flag de interrupção
  }
  VICVectAddr = 0x00000000;                   // Escreve sinal para final da interrupção
}
```

Fonte: adaptado de Trevor (2006, p. 72).

Figura 7 – Interrupção sem vetor

2.2 COMPUTADOR DE BORDO

Computadores de bordo foram projetados para facilitar a condução de veículos, informando ao motorista sobre possíveis falhas no automóvel, assim como dados importantes da situação atual do veículo, como velocidade, distância percorrida, rotações por minuto do motor, nível de combustível, temperatura do sistema de arrefecimento, carga da bateria, entre inúmeros outros.

Alguns computadores de bordo mais avançados realizam projeções de quilômetros possíveis a percorrer com a quantidade de litros restantes no tanque de combustível. Essa informação é obtida através da quantidade de quilômetros percorridos por litros de combustível consumidos (RISNIK, 2006a).

2.2.1 Sensores automotivos

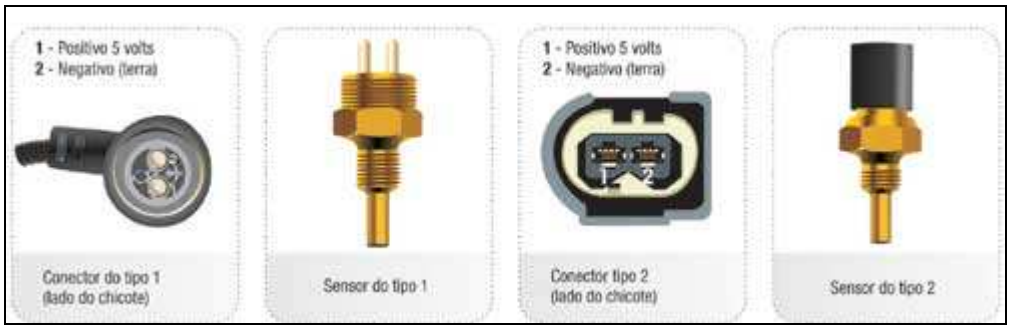
Sensores podem transformar uma grandeza do mundo exterior em um sinal elétrico, podendo assim obter dados reais de temperatura, pressão, luz, entre outros.

Os automóveis modernos possuem diversos sensores contribuindo desde controle do motor à segurança do passageiro. Os sensores estão presentes em cada canto do carro, monitorando e controlando o desempenho, a segurança e as operações básicas do veículo

(SUPERINTENDÊNCIA DA ZONA FRANCA DE MANAUS, 2006).

Um dos mais importantes sensores é o de temperatura da água que está posicionado no cabeçote ou bloco do motor. É utilizado basicamente no cálculo de tempo da injeção eletrônica e para evitar o superaquecimento do motor. Cada tipo de sensor pode ter uma característica própria de funcionamento e retorno das informações obtidas. Por exemplo, o sensor de temperatura da Mercedes-Benz, possui como padrão a saída de 5 volts quanto a temperatura for 0 graus centígrados, diminuindo a tensão proporcionalmente ao aumento da temperatura (OFICINA BRASIL, 2006).

A Figura 8 mostra o sensor de temperatura.



Fonte: Oficina Brasil (2006).

Figura 8 – Sensor de temperatura tipo 1 e tipo 2

O sensor de velocidade normalmente origina-se na caixa de transmissão do veículo e pode variar de um veículo para outro, que normalmente oscila entre 4 a 8 pulsos por giro de roda (RISNIK, 2006a).

A Figura 9 apresenta o sensor de velocidade.

Fonte: Risnik (2006a).

Figura 9 – Sensor de velocidade da Ford

No Quadro 1 são apresentados alguns tipos de sensores presentes no veículo.

Sensor	Tipo	Função
Velocidade	Pulsante - digital	Velocidade do veículo
RPM	Pulsante - digital	Quantidade de rotações do motor
Temperatura	Tensão - analógico	Temperatura do motor
Combustível	Tensão - analógico	Nível de combustível no tanque
Bateria	Tensão - analógico	Tensão da bateria do veículo

Quadro 1 – Tipos de sensores

2.3 *DISPLAY* LCD

Telas de cristal líquido estão se tornando cada vez mais comuns para a apresentação de dados, mesmo porque há determinadas situações nas quais seria inviável utilizar os convencionais monitores de tubo, que são grandes e pesados (PIROPO, 2006).

LCDs possuem baixo consumo de energia, são de fácil visualização e interpretação, sendo os mais comuns os LCDs alfanuméricos de 16 a 20 colunas com 1, 2 ou 4 linhas para texto. Possuem diversas cores, sendo de fundo verde o padrão encontrado no mercado. Existem modelos de LCD que possuem iluminação interna para melhor visualização em ambientes com pouca luminosidade.

Os LCDs possuem um microprocessador interno que controla os dados no *display*, facilitando assim a sua utilização em projetos eletrônicos. Já os tradicionais *displays* de 7 segmentos que são utilizados normalmente em calculadoras e muitos instrumentos digitais, são mais trabalhosos para implementar, pois cada *display* irá mostrar apenas um número, sendo necessários diversos *displays* para montar uma informação completa.

Segundo Braga (1989, p. 11), LCDs já existem a bastante tempo no mercado, inclusive com versões gráficas, porém ainda hoje, os LCDs alfanuméricos são os mais utilizados, devido, principalmente, ao custo e tamanho reduzidos comparado com os modelos mencionados.

A Figura 10 mostra um *display* de cristal líquido.

Fonte: Winstar Displays (2006).

Figura 10 – *Display* de cristal líquido WH1602L

A descrição de cada pino do LCD é demonstrado no Quadro 2.

Nr. Pino	Símbolo	Função
1	Vss	Terra (GND)
2	Vdd	Positivo(3 ou 5 volts)
3	VO	Ajuste de contraste
4	RS	1: Dados; 0: Instrução
5	R/W	1: Leitura; 0: Escrita
6	E	Habilitação do módulo LCD
7	DB0	Bit 0 de dados
8	DB1	Bit 1 de dados
9	DB2	Bit 2 de dados
10	DB3	Bit 3 de dados
11	DB4	Bit 4 de dados
12	DB5	Bit 5 de dados
13	DB6	Bit 6 de dados
14	DB7	Bit 7 de dados

Fonte: adaptado de Winstar Displays (2006).

Quadro 2 – Pinagem *display* WH1602

2.4 TRABALHOS CORRELATOS

Em Freese (2003) é apresentado um computador de bordo automotivo contendo funções como indicação de temperatura ambiente, velocidade veicular e situação de carga da bateria. As funções são controladas por um microcontrolador da família PIC. Este microcontrolador está interligado através da porta serial a um PC, que possui as funções de registro de excesso de velocidade e execução de músicas no formato MP3. O sistema pode ser operado através de um controle remoto, sendo mostradas as informações do computador de bordo e nome da música em execução, em um LCD. O software foi desenvolvido utilizando a linguagem de programação C. Para obter os dados de temperatura ambiente e velocidade, foram instalados novos sensores no veículo.

Risnik (2006a) descreve o produto denominado CCS-28, utilizando o microcontrolador 8051, o qual é um computador de bordo automotivo que possui as funções de velocidade atual, quatro alertas de velocidade programável, hodômetro parcial e total. Ainda o registro de velocidade máxima, horários de partida, chegada e quilometragem são disponibilizados. Cálculo de consumo de combustível, cronômetros e relógio de hora atual também são mostrados. O sistema pode ser configurado em qualquer veículo, conforme o sensor de velocidade presente. Caso não possua, será necessária a aquisição de um sensor.

Risnik (2006b) apresenta um contador de giros para qualquer tipo de veículo automotor, denominado SYS3. Utiliza o microcontrolador 8051 e mostra de forma digital em um LCD a quantidade de rotações por minuto do motor. Pode ser programado um alerta para quantidade de giros máximos, o qual acenderá uma lâmpada quando for atingido o limite programado. Permite utilização em veículos de 4, 6 ou 8 cilindros. Possui também indicador de bateria fraca e cronômetro.

3 DESENVOLVIMENTO DO SISTEMA

Esse capítulo está dividido em cinco seções. A primeira delas apresenta os requisitos funcionais e não funcionais do sistema a ser desenvolvido. A segunda seção apresenta uma visão geral do funcionamento do sistema. A terceira seção apresenta a especificação do sistema desenvolvido através de diagramas que o representam logicamente. A quarta seção, aborda alguns aspectos da implementação do sistema. A quinta seção apresenta os resultados e discussões do trabalho.

3.1 REQUISITOS PRINCIPAIS

Os requisitos funcionais de hardware são: receber como entrada os sinais analógicos dos sensores de temperatura, nível de combustível e tensão de bateria, além do sensor digital de velocidade e RPM; armazenar em uma memória EEPROM os parâmetros dos sensores, assim como o hodômetro total; disponibilizar os dados através de um LCD.

O requisito não funcional de hardware é: utilizar o microcontrolador ARM.

Os requisitos funcionais de software são: permitir a configuração do computador de bordo através de um PC interligado pela porta serial; transformar os dados lidos dos sensores em valores consistentes.

O requisito não funcional de software é: utilizar o compilador WinArm através da linguagem de programação C.

3.2 ESPECIFICAÇÃO

A especificação do sistema apresenta-se através do diagrama da UML, utilizando para tal os diagramas de casos de uso e diagrama de atividades.

3.2.1 Especificação do protótipo

Os diagramas da UML foram gerados com a utilização da ferramenta Enterprise Architect. A seguir são apresentados os diagramas de casos de uso e atividades.

3.2.1.1 Diagrama de casos de uso

O protótipo possui três casos de uso associados a um único ator: o usuário. Os casos de uso são: Usuário habilita modo de configuração; Usuário altera menu; e Usuário zera hodômetro.

O caso de uso Usuário habilita modo de configuração consiste na configuração inicial de alguns elementos fundamentais para o funcionamento do sistema, como sensores e variáveis.

O caso de uso Usuário altera menu consiste na alteração da visualização dos dados que serão mostrados no LCD.

O caso de uso Usuário zera hodômetro consiste na exclusão dos dados salvos para cada um dos quatro hodômetros, como a distância percorrida, velocidade média, velocidade máxima, consumo médio, máxima velocidade angular do motor por minuto e tempo decorrido em segundos.

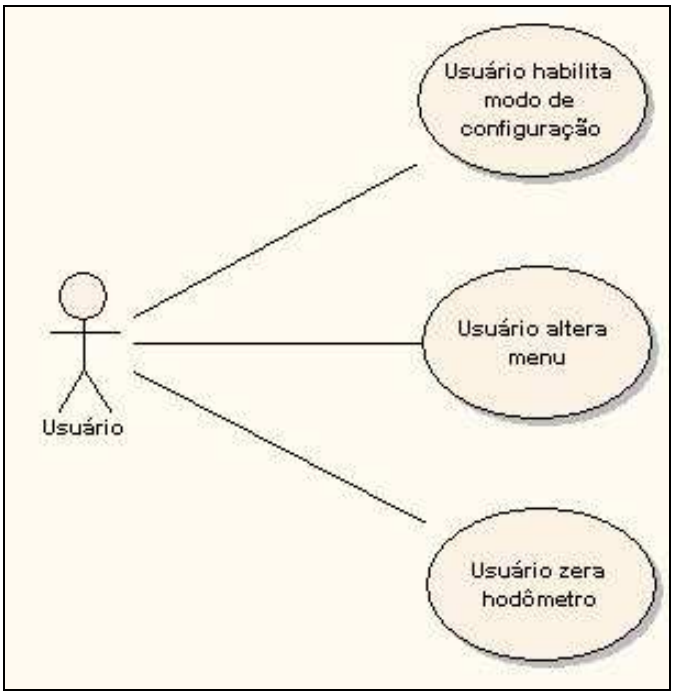


Figura 11 – Diagrama de casos de uso do sistema

3.2.1.2 Diagrama de atividades

No diagrama de atividades estão especificadas as funcionalidades do protótipo de forma geral. Na Figura 12 é apresentado o diagrama de atividades do protótipo.

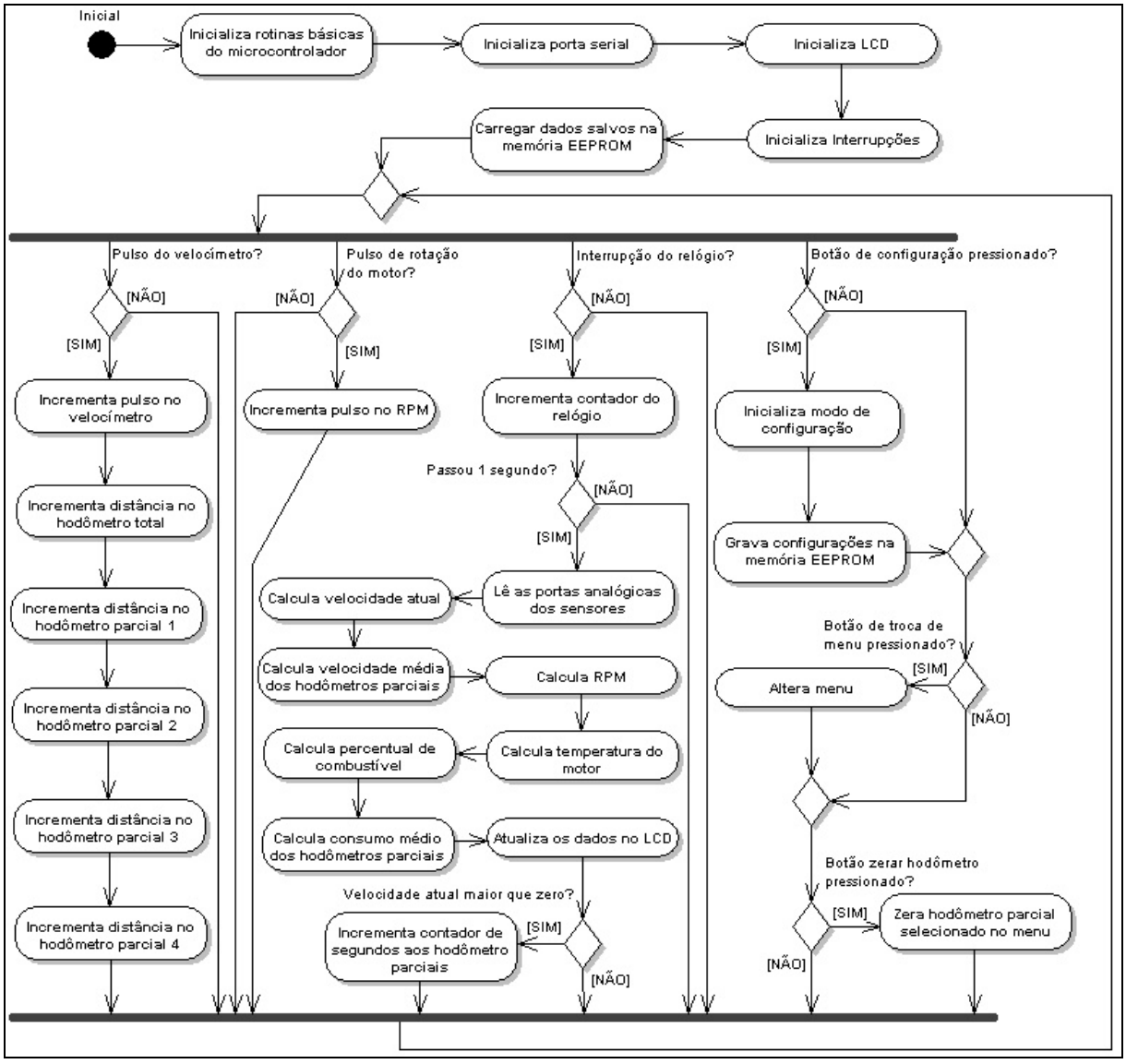


Figura 12 – Diagrama de atividades da ferramenta

A descrição detalhada do diagrama de atividades é apresenta no Quadro 3.

Atividade	Descrição
Inicializa rotinas básicas do microcontrolador	Inicializa as variáveis fundamentais para o funcionamento do microcontrolador, seta a frequência de *clock* do microcontrolador e define as portas de entrada e saída.
Inicializa porta serial	Seta os pinos de comunicação para a porta serial e habilita o serviço.
Inicializa LCD	Seta os pinos necessário para o LCD, envia os comandos de inicialização e escreve a mensagem de boas vindas.
Inicializa interrupções	Define e inicializa as interrupções do timer, velocímetro e RPM.
Carrega dados salvos na	Carrega da memória EEPROM os dados do hodômetro total e

memória EEPROM	configurações dos sensores.
Incrementa pulso no velocímetro	Incrementa variável contendo os pulsos gerados pelo sensor do velocímetro no período de um segundo.
Incrementa distância no hodômetro total	Baseado na distância por pulso do sensor do velocímetro, calcula a distância e soma ao hodômetro total.
Incrementa distância no hodômetro parcial 1	Baseado na distância por pulso do sensor do velocímetro, calcula a distância e soma ao hodômetro parcial 1.
Incrementa distância no hodômetro parcial 2	Baseado na distância por pulso do sensor do velocímetro, calcula a distância e soma ao hodômetro parcial 2.
Incrementa distância no hodômetro parcial 3	Baseado na distância por pulso do sensor do velocímetro, calcula a distância e soma ao hodômetro parcial 3.
Incrementa distância no hodômetro parcial 4	Baseado na distância por pulso do sensor do velocímetro, calcula a distância e soma ao hodômetro parcial 4.
Incrementa pulso no RPM	Incrementa variável contendo os pulsos das rotações do motor no período de um segundo.
Incrementa contador do relógio	Incrementa o contador do relógio utilizado para pausas e para o exato cálculo dos dados a cada segundo.
Lê as portas analógicas dos sensores	Obtêm os dados através das portas analógicas dos sensores da temperatura do motor, combustível e carga da bateria.
Calcula velocidade atual	Através dos pulsos gerados pelo sensor do velocímetro no período de um segundo, e pelos dados informados na configuração dos sensores, calcula a velocidade atual do veículo em kilômetros por hora. Verifica se velocidade é maior que o limite de alarme, imitindo um alerta visual.
Calcula a velocidade média dos hodômetros parciais	Através da distância percorrida nos hodômetros parciais de 1 a 4 dividido pelo tempo total do veículo em movimento em cada parcial, é calculada a velocidade média em kilômetros por hora.
Calcula RPM	Através dos pulsos gerados pelo sensor do RPM no período de um segundo, e pela configuração de cilindros do motor informado nas configurações, é calculada a velocidade angular do motor. Atualiza o RPM máximo de cada hodômetro parcial e verifica se RPM é maior que o limite de alarme, imitindo um alerta visual.
Calcula temperatura do motor	Através da tensão obtida do sensor de temperatura do sistema de arrefecimento e dos dados informados na configuração dos sensor, é calculada a temperatura em graus centígrados. Verifica se temperatura é maior que o limite de alarme, imitindo um alerta visual.
Calcula percentual de combustível	Através da tensão obtida do sensor de combustível e dos dados informados na configuração do sensor, é calculado o percentual de combustível restante no tanque.
Calcula consumo médio dos hodômetros parciais	Através da distância percorrida nos hodômetros parciais de 1 a 4 dividido pelo combustível consumido em cada parcial, é calculado o consumo médio do veículo em kilômetros por litro.
Atualiza os dados no LCD	Mostra os dados no LCD conforme o menu selecionado.
Incrementa contador de segundos aos hodômetros parciais	Incrementa variável contendo o tempo em segundos do veículo em movimento para cada hodômetro parcial.
Inicializa modo de configuração	O sistema entra em modo de configuração, listando os menus através da porta serial.
Grava configurações na memória EEPROM	É gravado na memória EEPROM o valor do hodômetro total e as configurações dos sensores.
Altera menu	Altera o menu de visualização dos dados no LCD.
Zera hodômetro parcial selecionado no menu	Zera os dados do hodômetro parcial selecionado no menu.

Quadro 3 – Descrição das atividades

3.2.1.2.1 Atividade de configuração

Dentre as atividades apresentadas no diagrama ilustrado na Figura 12, a atividade de configuração é a única que necessita estar conectada a um PC através da porta serial para ser executada, necessitando de um melhor detalhamento, conforme diagrama de atividades apresentado na Figura 13.

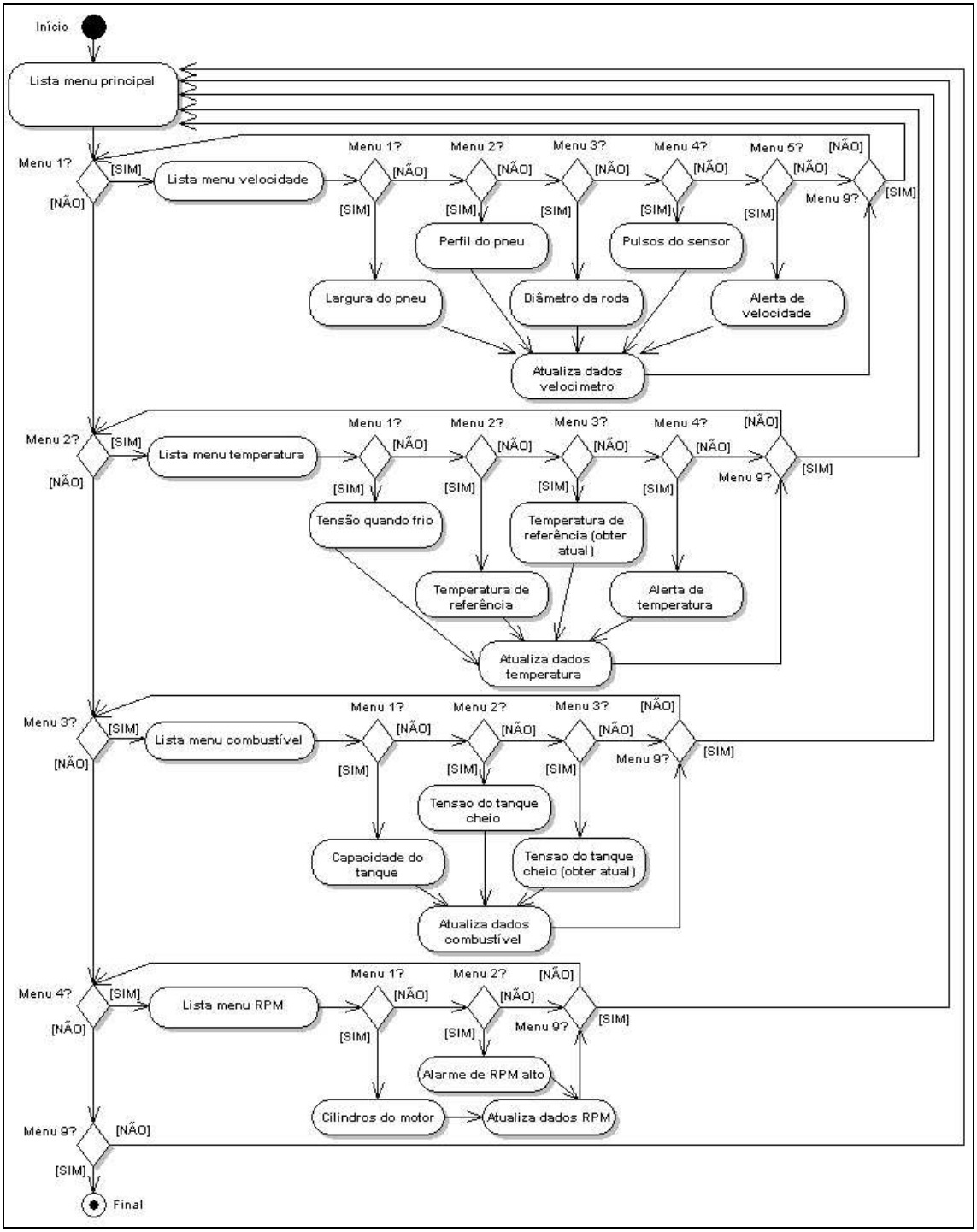


Figura 13 – Diagrama de atividades da configuração

A descrição detalhada do diagrama de atividades é apresenta no Quadro 4.

Atividade	Descrição
Lista menu principal	Lista o menu principal de configuração através da porta serial.
Lista menu velocidade	Lista o menu de configuração de velocidade.
Largura do pneu	Usuário informa a largura do pneu em milímetros.
Perfil do pneu	Usuário informa o percentual do perfil do pneu em relação a largura do mesmo.
Diâmetro da roda	Usuário informa o diâmetro da roda em polegadas.
Pulsos do sensor	Usuário informa a quantidade de pulsos por giro de roda emitido pelo sensor de velocidade.
Alerta de velocidade	Usuário informa a velocidade em kilômetros por hora para emissão de alerta quando ultrapassado.
Atualiza dados velocímetro	Atualiza os dados do velocímetro para utilização no sistema.
Lista menu temperatura	Lista o menu de configuração de temperatura.
Tensão quando frio	Usuário informa a tensão do sensor em volts quando o motor estiver frio.
Temperatura de referência	Usuário informa uma temperatura em graus centígrados e sua tensão equivalente em volts para referência no cálculo da temperatura.
Temperatura de referência (obter atual)	Usuário informa uma temperatura em graus centígrados pegando como base a tensão atual do sensor de temperatura.
Alerta de temperatura	Usuário informa uma temperatura em graus centígrados para emissão de alarme quando for ultrapassada.
Atualiza dados temperatura	Atualiza os dados da temperatura para utilização no sistema.
Lista menu combustível	Lista o menu de configuração de combustível.
Capacidade do tanque	Usuário informa a capacidade de litros do tanque de combustível.
Tensão do tanque cheio	Usuário informa a tensão em volts do sensor de combustível quando o tanque estiver cheio.
Tensão do tanque cheio (obter atual)	Será assumida a tensão atual do sensor de combustível como estando com o tanque cheio.
Atualiza dados combustível	Atualiza os dados do combustível para utilização no sistema.
Lista menu RPM	Lista os menus de configuração do RPM.
Cilindros do motor	Usuário informa a quantidade de cilindros do motor.
Alarme de RPM alto	Usuário informa um RPM para emissão de alerta quando ultrapassado.
Atualiza dados RPM	Atualiza os dados do RPM para utilização no sistema.

Quadro 4 – Descrição das atividades de configuração

3.3 IMPLEMENTAÇÃO

A implementação do sistema está divida em duas seções. Na primeira seção, apresentam-se algumas considerações sobre as técnicas e ferramentas utilizadas para a sua implementação. Na segunda seção, apresenta-se a operacionalidade e o funcionamento da implementação.

3.3.1 Técnicas e ferramentas utilizadas

O software do protótipo foi implementado utilizando a linguagem de programação C no ambiente de desenvolvimento WinArm (THOMAS, 2006). Através do microcontrolador ARM foram utilizados os recursos de entrada e saída de portas, leitura de portas analógicas, interrupção interna de *timer*, interrupção externa, mostrado os dados em LCD, envio e recepção de dados pela porta serial e gravação de dados na memória EEPROM. Na Figura 14 é apresentado o ambiente de desenvolvimento do WinArm.

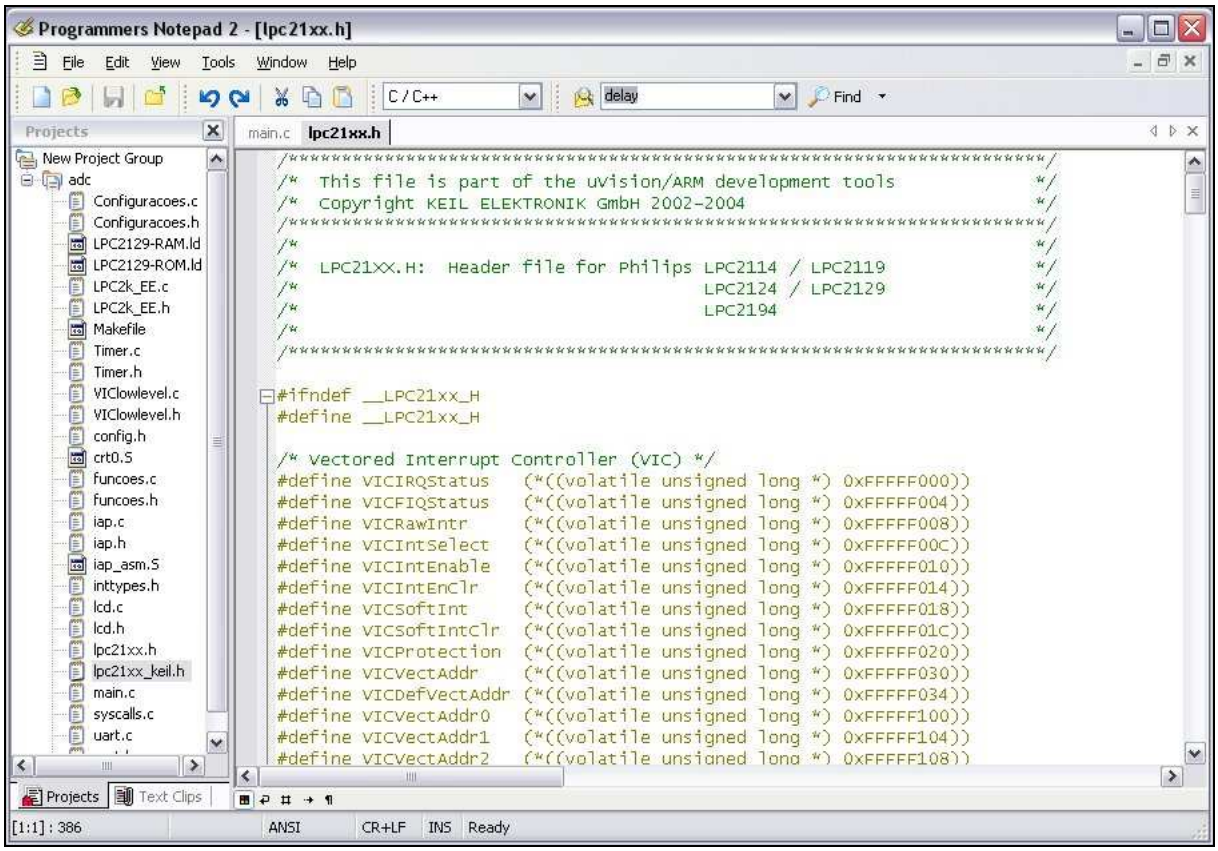


Figura 14 – Ambiente de desenvolvimento do WinArm

Para gravar o projeto compilado no microcontrolador ARM, foi utilizado o aplicativo da Philips LPC2000 *Flash Utility* (PHILIPS ELETRONICS, 2006), o qual envia os dados para o microcontrolador através da porta serial. Na Figura 15 é apresentado o utilitário da Philips.

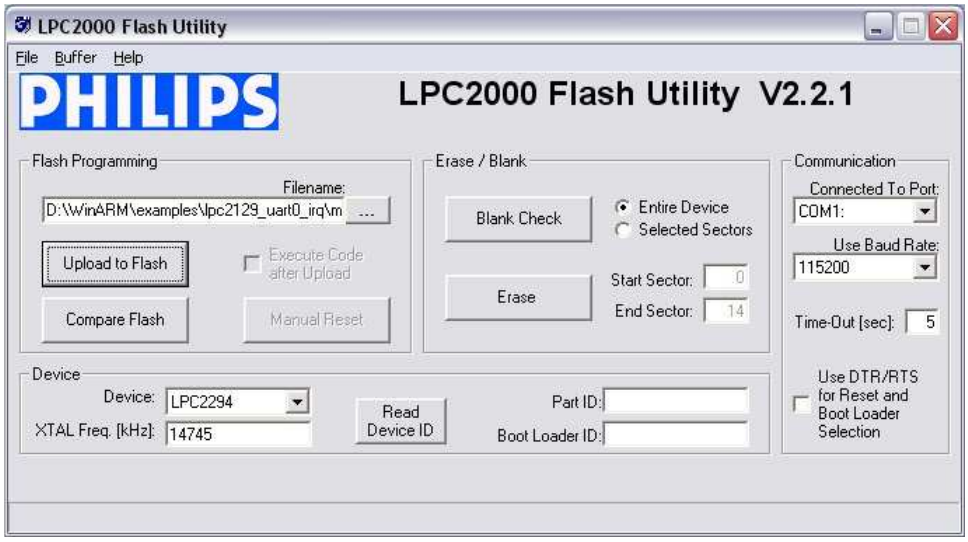


Figura 15 – LPC2000 *Flash Utility*

O protótipo foi inicialmente projetado utilizando o ambiente de simulação do Proteus (LABCENTER ELETRONICS, 2006), o qual possui diversos componentes eletrônicos sendo possível simular todas as funcionalidades do projeto.

Na Figura 16 é apresentado o ambiente de simulação do Proteus.

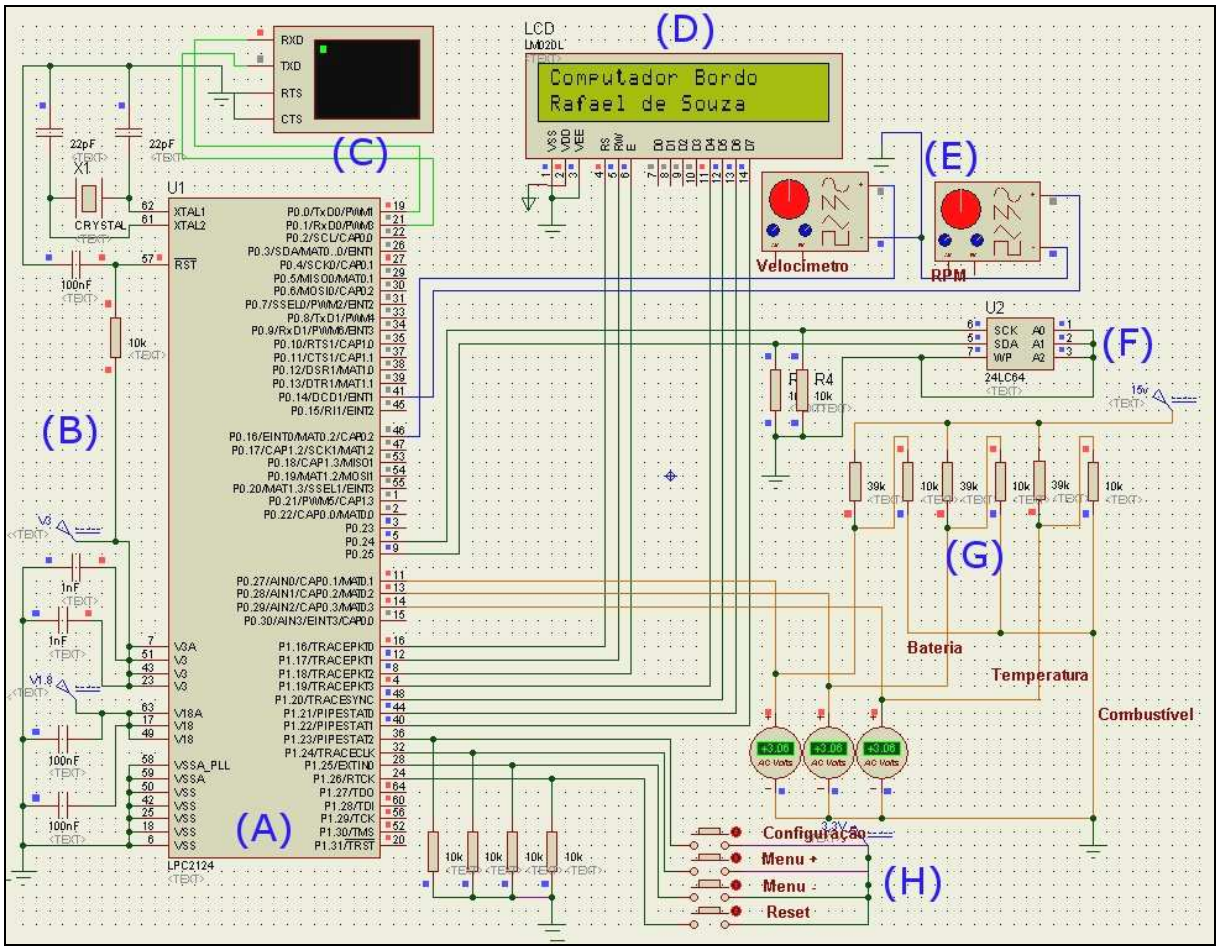


Figura 16 – Ambiente de simulação do Proteus

A descrição dos componentes acima representados dar-se-á da seguinte forma:

a) microcontrolador ARM;
b) componentes básicos necessários para o funcionamento do microcontrolador;
c) simulador de um terminal ligado a porta serial do microcontrolador;
d) simulador de LCD;
e) geradores de *clock* ligados nos pinos de interrupção externa do microcontrolador para simular o velocímetro e RPM;
f) memória EEPROM para a gravação permanente dos dados;
g) circuito para simulação das entradas de tensão dos sensores da temperatura, combustível e bateria;
h) botões para interação do usuário com o protótipo.

Para o desenvolvimento das funções de cálculos como a de velocidade veicular, foram utilizados cálculos de mecânica física, como a velocidade escalar média, no qual é a distância percorrida em um período de tempo, obtendo-se a velocidade média. Como os dados são atualizados a cada segundo no *display*, temos a velocidade média percorrida a cada um segundo. A Figura 17 apresenta a função de cálculo da velocidade veicular.

```
#define pi 3,14159265358979323846264338327795
// Variáveis gravadas na EEPROM
unsigned long vHodometro;    // Distância total em centímetros
unsigned char vTamanhoAro;   // Tamanho do aro em polegadas
unsigned char vPneuPerfil;   // Perfil do pneu em percentual
unsigned int  vPneuLargura;  // Largura do pneu em milímetros
unsigned char vPulsosGiroRoda;
// Variáveis de uso global
unsigned char vVelocidade;   // Velocidade atual em kilômetros por hora
unsigned long vHodometro1;   // Distância parcial 1 em centímetros
unsigned long vHodometro2;   // Distância parcial 2 em centímetros
unsigned long vHodometro3;   // Distância parcial 3 em centímetros
unsigned long vHodometro4;   // Distância parcial 4 em centímetros

void CalculaVelocidade(void)
{
  float vDistanciaPulso; // Distância em centímetros percorrida por pulso do sensor de velocidade
  vDistanciaPulso = ((vTamanhoAro * 2.54 + 2 * vPneuPerfil * vPneuLargura / 1000) * pi) / vPulsosGiroRoda;
  // Pega a quantidade de pulsos gerado em 1 segundo, multiplica a distância percorrida em cm
  // o resultado está em centímetros por segundo, converter para Km/h, o resultado da equação é 0.036
  vVelocidade = vPulsosVel * vDistanciaPulso * 0.036;
  // Atualiza o hodômetro total
  vHodometro = vHodometro + vPulsosVel * vDistanciaPulso;
  // Atualiza os hodômetros parciais
  vHodometro1 = vHodometro1 + vPulsosVel * vDistanciaPulso;
  vHodometro2 = vHodometro2 + vPulsosVel * vDistanciaPulso;
  vHodometro3 = vHodometro3 + vPulsosVel * vDistanciaPulso;
  vHodometro4 = vHodometro4 + vPulsosVel * vDistanciaPulso;
}
```

Figura 17 – Cálculo da velocidade veicular

Para o cálculo do percentual de combustível, é realizada uma leitura da tensão atual do sensor de combustível e comparada com a tensão informada na configuração para quando o tanque estiver cheio.

A Figura 18 apresenta a função de cálculo do percentual de combustível restante no tanque do veículo.

```
#define VALORPORTENSAO 1023/15        // Valor equivalente a cada 1 volt de tensão
                                      // Foi definido como 15 volts por questão de segurança
// Variáveis gravadas na EEPROM
float vTensaoTanqueCheio;           // Tensão do tanque quando estiver cheio
float vTensaoTanqueVazio;           // Tensão do tanque quando estiver vazio

// Lê as portas analógicas
unsigned short RetornaAD(unsigned char ch)
{
  unsigned int i;
  ADCR = 0x00200300 | ch;             // Seta parâmetros do A/D e seleciona o canal A/D desejado
                                      // Canal: 1=AN0, 2=AN1, 4=AN2, 8=AN3, 16=AN4, 32=AN5, 64=AN6, 128=AN7
  ADCR |= 0x01000000;                 // Inicia conversão do A/D
  do
  {
    i = ADDR;                         // Lê os dados do registrador do A/D
  } while ((i & 0x80000000) == 0);    // Fica em loop até terminar a conversão e 'i' ter o valor do A/D
  return (i >> 6) & 0x03FF;           // Faz um shift de 6 bits para a direita e pega os 10 primeiros bits
                                      // que é o valor do A/D (0 a 1023)
}

// Calcula o percentual de combustível restante no tanque
unsigned int PercentualCombustivel(void)
{
  unsigned int vCombustivelAtual;
  float vValorTanqueCheio;
  // Calcula o valor equivalente ao A/D do tanque cheio
  vValorTanqueCheio = vTensaoTanqueCheio * VALORPORTENSAO;
  vValorTanqueVazio = vTensaoTanqueVazio * VALORPORTENSAO;
  // Retorna o valor A/D do pino ADTEMPERATURA (0 a 1023)
  vCombustivelAtual = RetornaAD(ADTEMPERATURA);
  // Retorna o percentual de combustível restante no tanque
  return (100 / (vValorTanqueCheio - vValorTanqueVazio) * (vCombustivelAtual - vValorTanqueVazio));
}
```

Figura 18 – Cálculo do percentual de combustível no tanque do veículo

3.3.2 Operacionalidade da implementação

Ao iniciar o protótipo, ele já estará em modo de execução, porém é necessário configurá-lo corretamente para funcionamento no veículo. Em modo normal de execução ao pressionar o botão de configuração, o sistema entrará em modo de configuração, porém deve-se previamente conectar o protótipo a um PC através da porta serial e utilizar um software de emulação de terminal, como o Hyper Terminal do Windows.

A configuração do sistema ao invés de ser feita pela porta serial poderia ser através do LCD e um teclado ligado ao protótipo, porém não seria viável e prático devido ao espaço reduzido de caracteres no LCD, desta forma optou-se pelo uso do terminal.

Para abrir o Hyper Terminal do Windows, o mesmo pode ser acessado diretamente pelo executar, digitando “hypertrm” conforme Figura 19.

Figura 19 – Executando o Hyper Terminal

Para que o terminal e o microcontrolador possam conversar pela porta serial, os mesmo devem estar configurados para trabalhar na mesma velocidade, a qual é medida em bits por segundo. O Hyper Terminal deverá ser configurado conforme a Figura 20.

Figura 20 – Configuração do Hyper Terminal

Inicialmente será impresso no terminal o menu principal que o usuário deseja modificar de acordo com as características do veículo utilizado . Na Figura 21 é apresentada o menu principal de configurações.

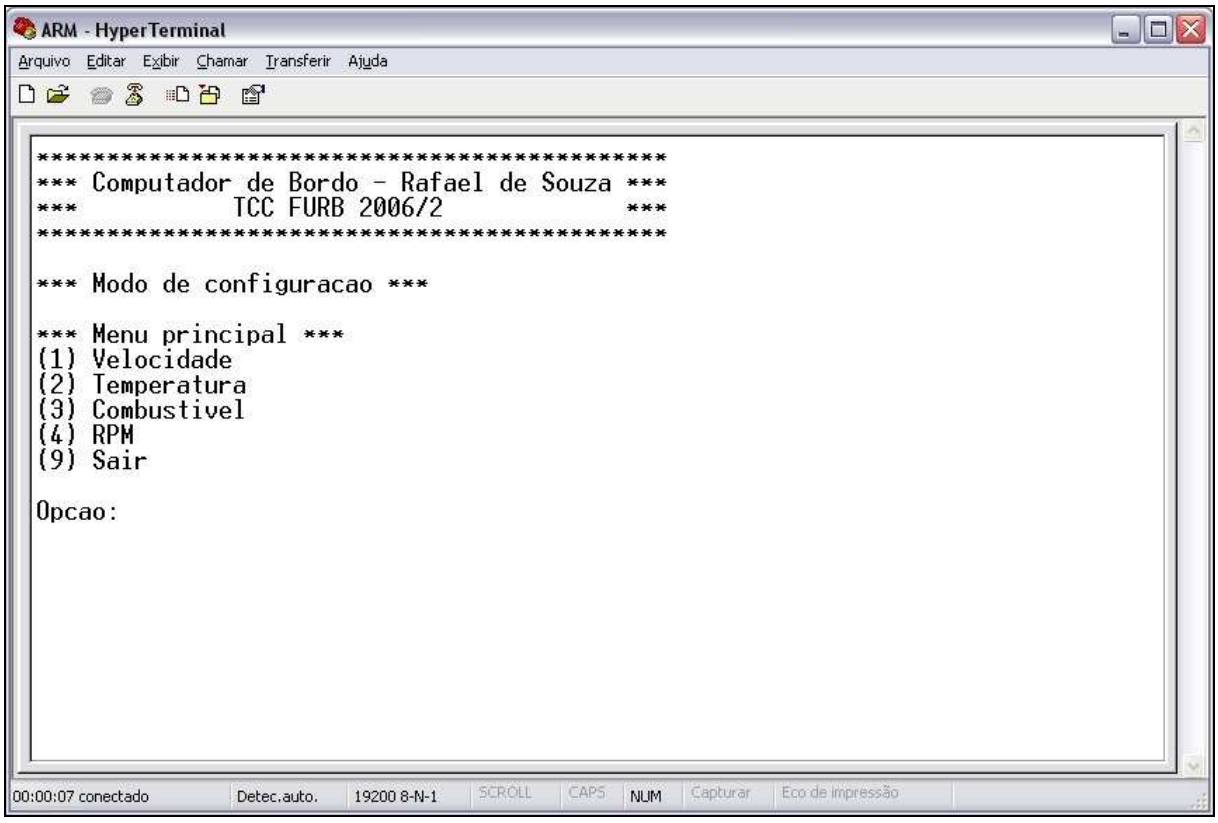


Figura 21 – Menu principal de configurações

Na opção velocidade poderá ser informada os dados conforme Figura 22.

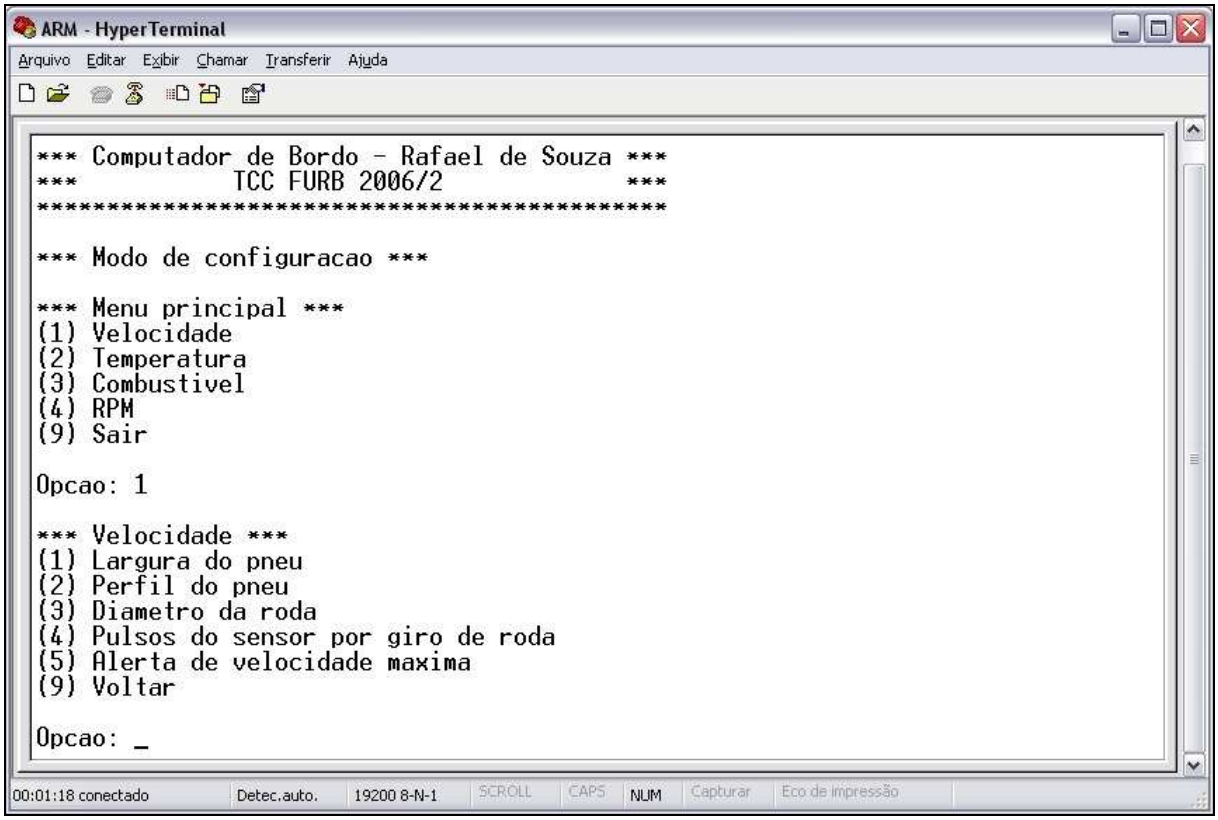


Figura 22 – Menu velocidade

Na opção 1 deverá ser informada a largura do pneu em milímetros. Na opção 2 deverá ser informada o perfil do pneu, no qual é o percentual de largura equivalente a altura. Na opção 3 deverá ser informado o diâmetro da roda em polegadas. Os dados das três primeiras opções são encontrados impressos no pneu do veículo, exemplo “195/50 R15”, onde a primeira informação é a largura, seguida do perfil e por último o diâmetro.

Na opção 4 deverá ser informada a quantidade de pulsos emitidos por giro de roda, normalmente variam entre 4 a 8 pulsos. Na opção 5 poderá ser definido um valor para o alerta de velocidade máxima, toda vez que ultrapassar o valor estipulado, será emitido um alarme visual indicando o excesso.

As configurações atualmente gravadas na memória EEPROM são mostradas no momento de alterar qualquer um dos dados conforme Figura 23.

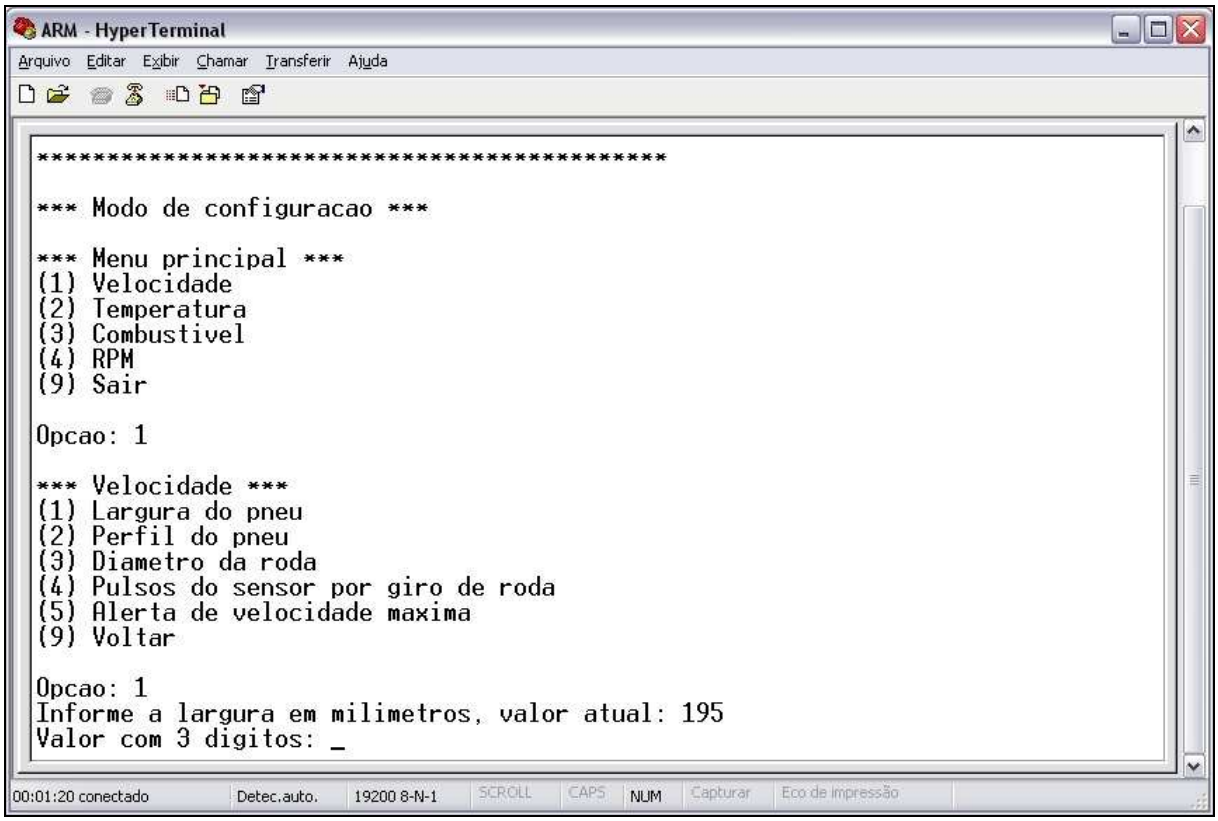


Figura 23 – Configuração gravada na memória EEPROM

Na opção temperatura poderá ser informado os dados conforme Figura 24.

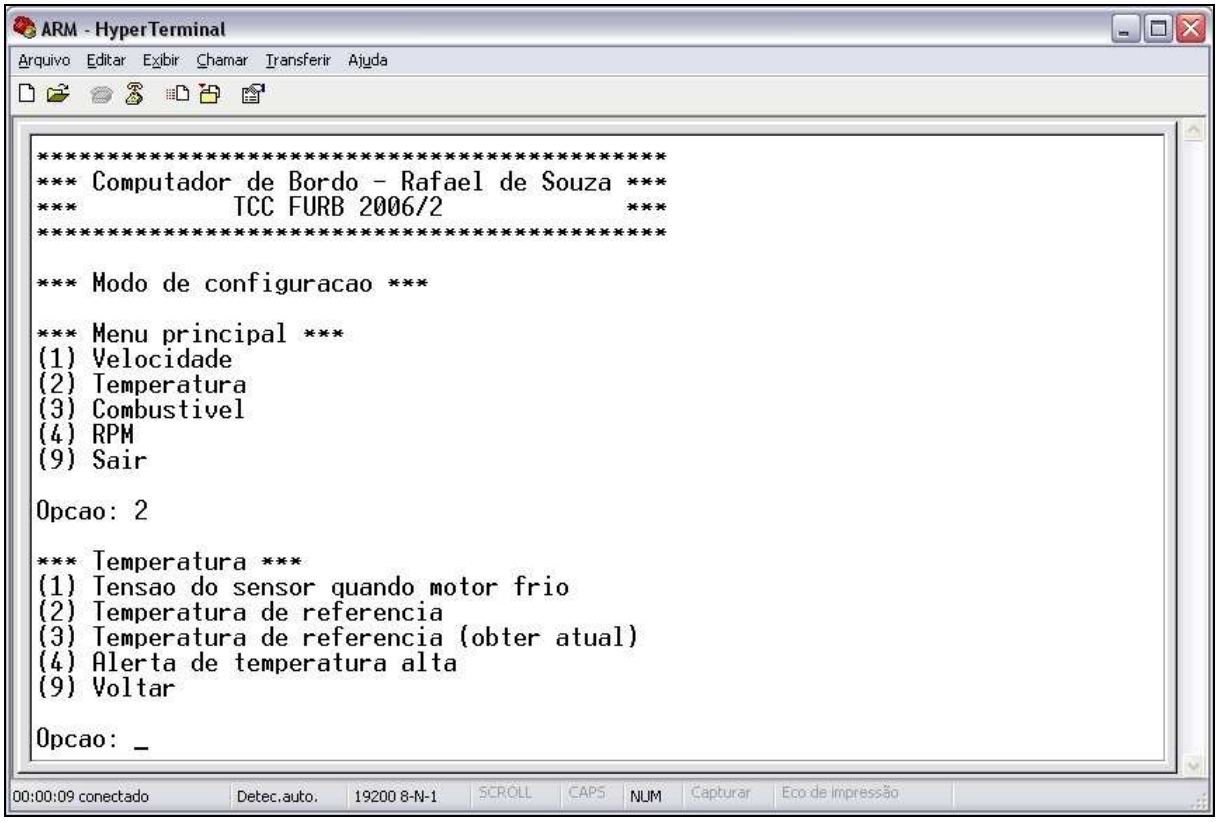


Figura 24 – Menu temperatura

Na opção 1 deverá ser informada a tensão do sensor de temperatura quando o motor

estiver frio, normalmente é de 5 volts. Na opção 2 deverá ser informada a temperatura de referência em graus centígrados, o qual é fundamental para o cálculo da temperatura correta, após informar a temperatura será solicitada a tensão em volts do sensor para a temperatura informada, por exemplo, a 90 graus centígrados o sensor retornará 1,18 volts. Na opção 3 poderá ser informada uma temperatura de referência em graus centígrados, obtendo a tensão atual retornada pelo sensor em volts, sendo necessário apenas informar a temperatura atual do motor. Na opção 4 poderá ser definido um valor para o alerta de temperatura máxima, sendo emitido um alerta visual sempre que exceder o limite estipulado.

Na opção combustível poderá ser informado os dados conforme Figura 25.

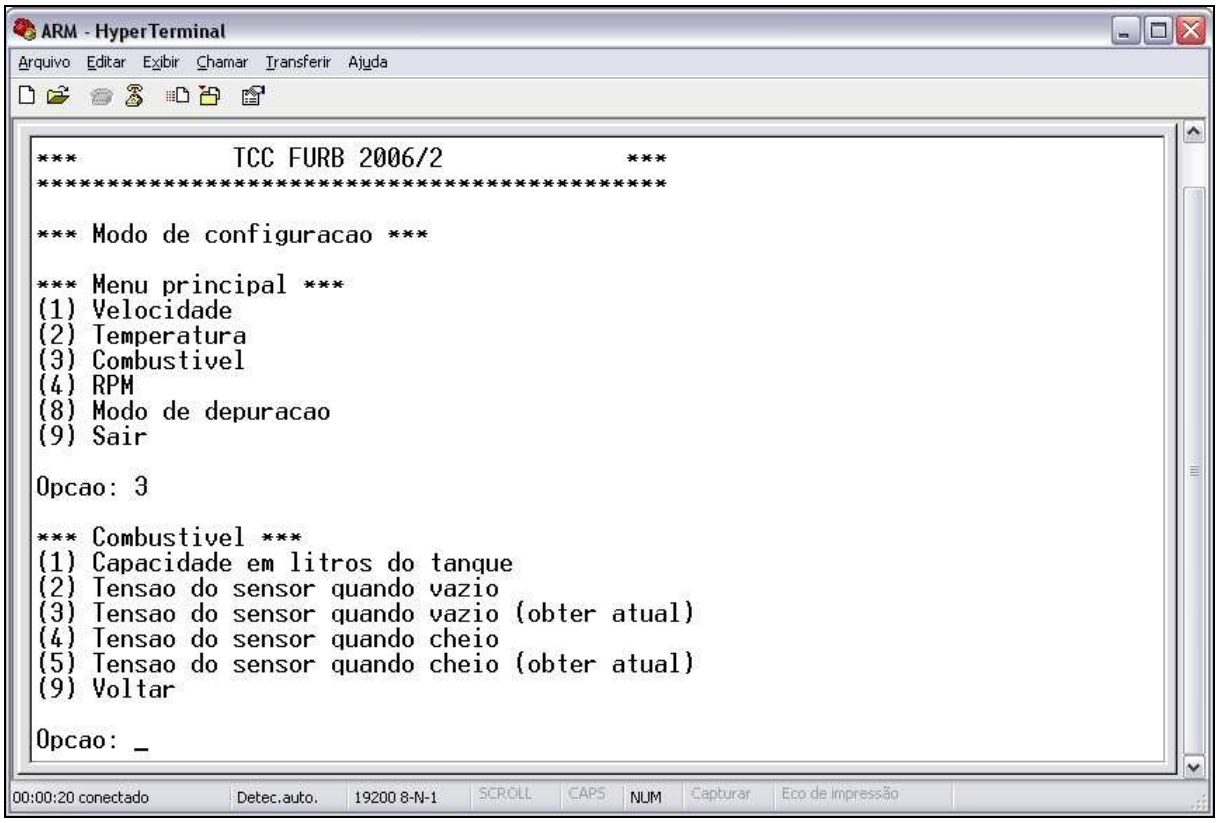


Figura 25 – Menu combustível

Na opção 1 deverá ser informada a capacidade em litros do tanque de combustível conforme consta no manual do veículo, sendo de fundamental importância para o cálculo correto do consumo médio. Na opção 2 deverá ser informada a tensão do sensor quando o tanque estiver vazio com sua capacidade mínima. Na opção 3 poderá ser definido para obter a tensão atual do sensor, assumindo o estado atual como estando com o tanque vazio. Na opção 4 deverá ser informada a tensão do sensor quando o tanque estiver cheio com sua capacidade máxima. Na opção 5 poderá ser definido para obter a tensão atual do sensor, assumindo o estado atual como estando com o tanque cheio.

Na opção RPM poderá ser informado os dados conforme Figura 26.

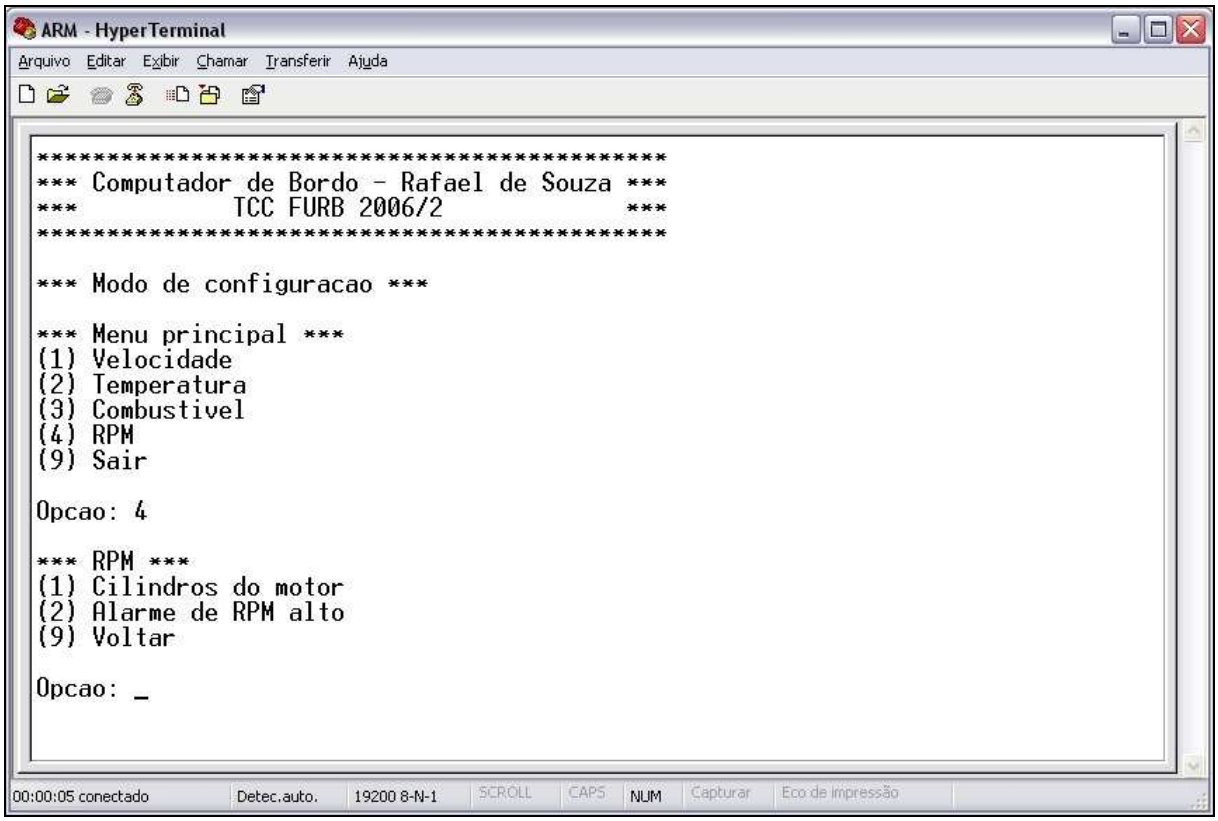


Figura 26 – Menu RPM

Na opção 1 deverá ser informado a quantidade de cilindros do motor. Na opção 2 poderá ser definido um valor para o alerta de RPM alto, sendo emitido um alerta visual sempre que exceder o limite estipulado. Ao sair do modo de configuração, o sistema voltará para o seu modo normal de execução.

Ao inicializar o computador de bordo irá aparecer no LCD a tela de saudação conforme Figura 27 e em seguida rapidamente o menu em que o sistema se encontra, sendo um total de dez menus, no qual o usuário poderá navegar entre eles pressionando os botões de alteração mais e menos.

Figura 27 – Tela de saudação

Na Figura 28 é apresentada a tela indicando rapidamente o menu 1.

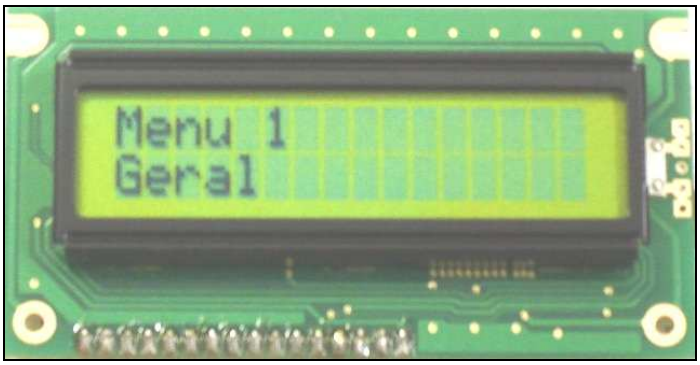


Figura 28 – Informação de menu 1

Em seguida irá apresentar os dados do menu 1 que são:

a) velocidade em kilômetros por hora, denominado “V”;

b) distância total percorrida em kilômetros, denominado “H”;

c) RPM, denominado “R”;

d) temperatura do motor em graus centígrados, denominado “T”.

Na Figura 29 é apresentado os dados do menu 1.

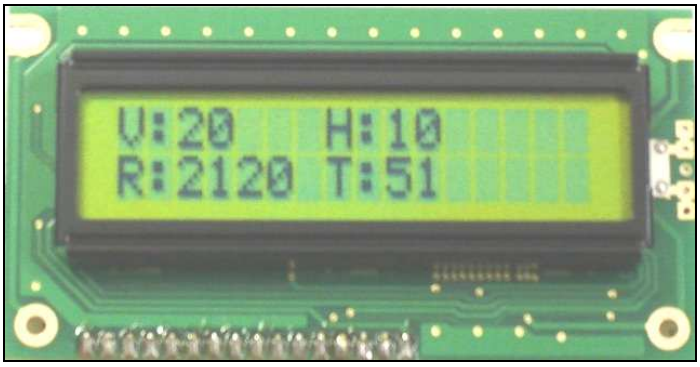


Figura 29 – Dados do menu 1

Ao alterar para o menu 2 irá apresentar rapidamente a tela indicando o menu selecionado conforme Figura 30.

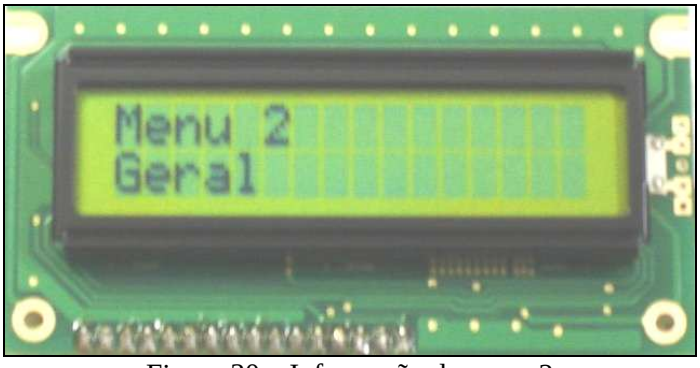


Figura 30 – Informação de menu 2

Em seguida irá apresentar os dados do menu 2 que são:

a) velocidade em kilômetros por hora, denominado “V”;

b) carga da bateria em volts, denominado “B”;

c) percentual de combustível restante no tanque, denominado “C”;

d) temperatura do motor em graus centígrados, denominado “T”.

Na Figura 31 é apresentado os dados do menu 2.

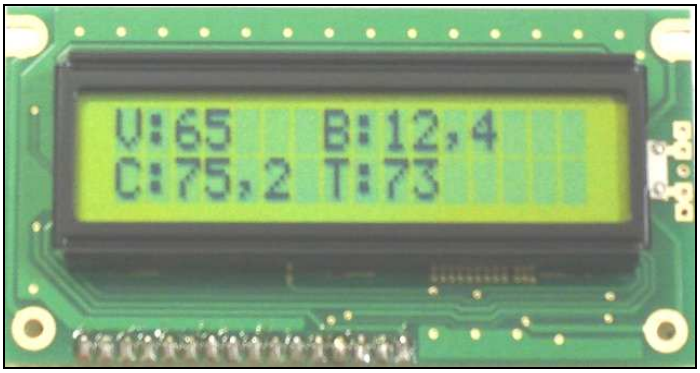


Figura 31 – Dados do menu 2

Ao alterar para o menu 3 irá apresentar rapidamente a tela indicando o menu selecionado conforme Figura 32.

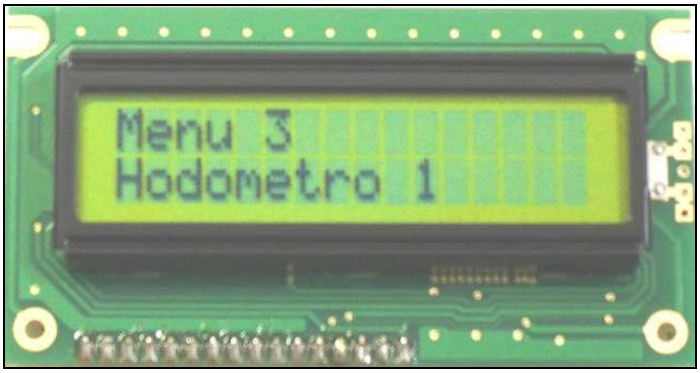


Figura 32 – Informação de menu 3

Em seguida irá apresentar os dados do menu 3 referentes ao hodômetro 1 que são:

a) velocidade em kilômetros por hora, velocidade média e velocidade máxima, denominado “V”;

b) distância percorrida do hodômetro 1 em kilômetros, denominado “H1”;

c) projeção de kilômetros possível com o combustível restante, denominado “P”.

Na Figura 33 é apresentado os dados do menu 3.

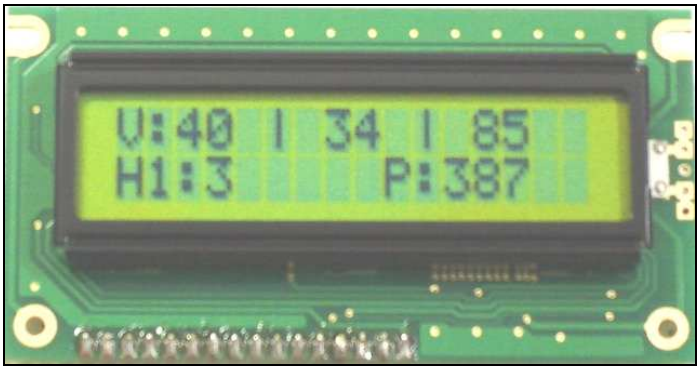


Figura 33 – Dados do menu 3

Ao alterar para o menu 4 irá apresentar rapidamente a tela indicando o menu selecionado conforme Figura 34.

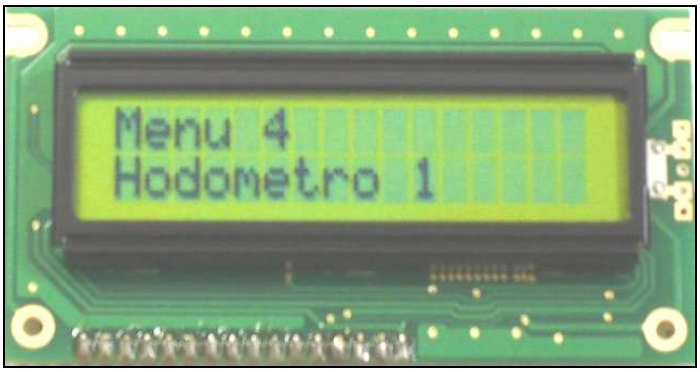


Figura 34 – Informação de menu 4

Em seguida irá apresentar os dados do menu 4 referentes ao hodômetro 1 que são:

a) velocidade em kilômetros por hora, denominado “V”;
b) tempo decorrido do veículo em movimento no formato hora, minuto e segundo, denominado “T”;
c) consumo médio de combustível em kilômetros por litro, denominado “CM”;
d) RPM máximo alcançado, denominado “RM”.

Na Figura 35 é apresentado os dados do menu 4.

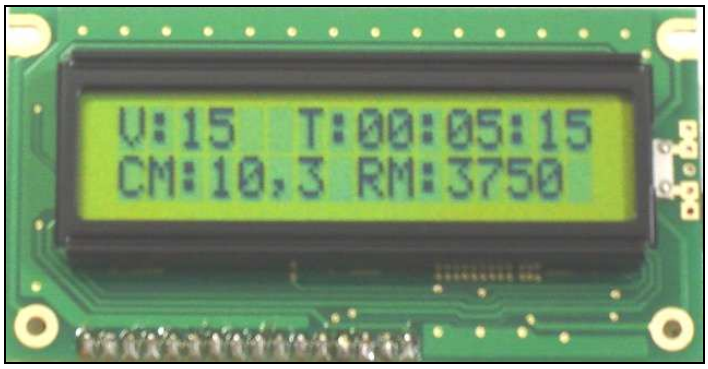


Figura 35 – Dados do menu 4

Os menus seguintes são a repetição dos menus 3 e 4 do hodômetro 1, porêm mostrando os dados dos hodômetros 2, 3 e 4 respectivamente.

A seguir na Figura 36 é apresentado o hardware do protótipo e seus componentes.

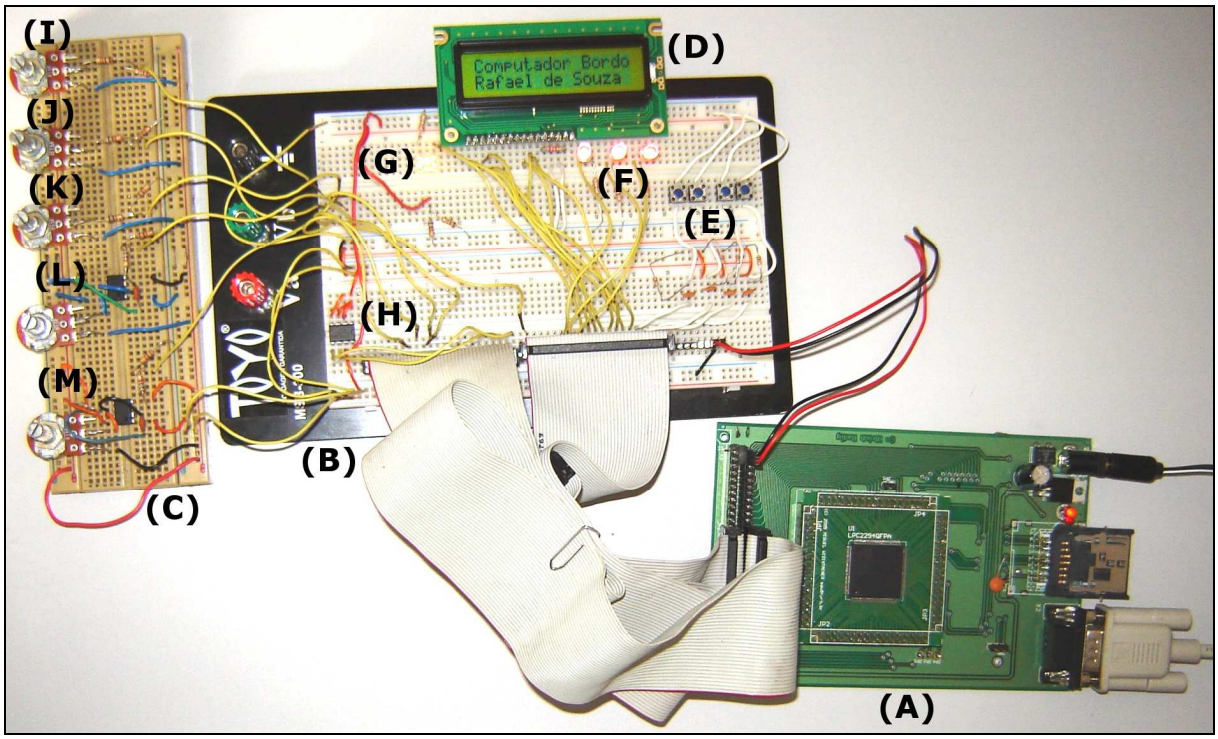


Figura 36 – Hardware do protótipo

A descrição dos componentes acima representados dar-se-á da seguinte forma:

a) circuito do microcontrolador ARM, projetado por Ulrich (ULRICH, 2006) e disponibilizado pelo professor e orientador Miguel Wisintainer e professor Antonio Carlos Tavares.
b) grupo de componentes do computador de bordo;
c) grupo de componentes utilizados para simular os sensores do veículo;
d) LCD para a apresentação dos dados ao usuário;
e) botões de interação do usuário com o computador de bordo, sendo eles: menu anterior, próximo menu, zerar hodômetro e modo de configuração;
f) luzes para indicação de alertas, sendo elas: temperatura alta, limite de velocidade ultrapassado e limite de RPM alcançado;
g) luz indicadora de funcionamento do sistema;
h) memória EEPROM para gravação permanente dos dados;
i) circuito de simulação de tensão do sensor de temperatura do sistema de arrefecimento do veículo;
j) circuito de simulação de tensão do sensor de combustível;
k) circuito de simulação de tensão da bateria;
l) circuito de simulação de pulsos do sensor de velocidade;
m) circuito de simulação de pulsos das rotações por minuto do motor.

3.4 RESULTADOS E DISCUSSÃO

Considerando a proposta do protótipo, obtiveram-se bons resultados. O microcontrolador ARM apresentou ótimo desempenho, realizou corretamente a coleta de dados dos sensores, transformando-os em dados consistentes como esperado. Apesar do modo de configuração ser através da porta serial, o protótipo pode ser facilmente configurado mesmo em utilização no veículo através de um computador portátil. Apesar de inicialmente parecer complicada a sua configuração, a mesma normalmente terá que ser alterada apenas uma única vez após a instalação no veículo, pois os dados não mais precisarão ser alterados.

Fazendo um comparativo com o computador de bordo de Freese (Freese, 2003), o presente trabalho demonstra ao usuário mais dados em relação ao veículo enquanto o do Freese está mais voltado a execução de músicas no formato MP3.

Realizando um comparativo com o computador de bordo CCS-28 (RISNIK, 2006a) e SYS3 (RISNIK, 2006b), o presente trabalho possibilita uma grande parte de suas funcionalidades em um único projeto como velocidade atual, alerta de velocidade máxima, consumo de combustível, RPM e carga da bateria.

No decorrer do trabalho foram encontradas algumas dificuldades, sendo a maior delas a escassez de material referente ao microcontrolador ARM e poucos exemplos de códigos desenvolvidos na linguagem C, muitas vezes não eram compatíveis com o ambiente WinArm, tendo que ser adaptados quando possível e simulados no Proteus para verificar seu perfeito funcionamento.

O microcontrolador ARM foi inserido recentemente no simulador Proteus, possuindo ainda algumas falhas, causando alguns erros isolados na simulação, exigindo um tempo extra para contornar a situação. O Proteus não possui microcontroladores da família LPC22XX utilizado no projeto, somente da família LPC21XX, sendo inferior com um número reduzido de pinos, possuindo apenas os grupos de pinos P0 e P1, sendo que no projeto são usados os grupos P2 e P3, pois foram estes os pinos disponibilizados no circuito de Ulrich (ULRICH, 2006).

4 CONCLUSÕES

Este trabalho apresentou um protótipo para auxiliar os usuários de automóveis, informando os dados do mesmo de forma digital e mais precisa em relação aos mostradores analógicos convencionais, apontando para uma nova tendência que vem crescendo no mercado automobilístico.

Dentre as principais vantagens do sistema desenvolvido, destaca-se o emprego do microcontrolador ARM, o qual é novo no mercado, tendo uma ótima relação de custo benefício, porém com pouco material disponível a respeito, exigindo um esforço extra para o seu entendimento, mas proporcionando o resultado esperado. O TCC será uma fonte para iniciantes em ARM no que se refere a programação, pois o código fonte está previamente documentado.

A desvantagem que se pode citar é que o protótipo deverá ser instalado em um veículo por um eletricista profissional da área, pois o mesmo deverá saber identificar os fios corretos dos sensores sem danificar a integridade original do sistema elétrico do veículo.

4.1 EXTENSÕES

Este trabalho pode ser continuado através da implementação de um *display* LCD gráfico, podendo organizar melhor a apresentação dos dados ao usuário, reduzindo a quantidade de menus e melhorando a estética, podendo até criar um menu de configurações utilizando o próprio LCD, sem a necessidade de utilizar a porta serial para isto.

Poderiam ser inseridos sensores de indicação de aproximação de objetos do veículo, muito úteis para facilitar no estacionamento do mesmo, implementação de um sistema de alarme anti-furto, assim como a instalação de um fototransistor para ligação automática de faróis em ambientes escuros.

Poderia ser implementado um sistema inteligente de detecção de fadiga do motorista, com uma câmera focalizando os olhos do mesmo, emitindo um alerta caso venha a adormecer enquanto conduz o veículo.

Como o microcontrolador ARM possui um ótimo desempenho, poderia ser criado um tocador de músicas no formato MP3, assim como um sistema integrado de GPS.

REFERÊNCIAS BIBLIOGRÁFICAS

ASSOCIAÇÃO BRASILEIRA DE MONITORAMENTO E CONTROLE ELETRÔNICO DE TRÂNSITO. **Os novos conceitos**. São Paulo, 2003. Disponível em: <http://www.abramcet.com.br/home/old/hist_conceitos.shtml>. Acesso em: 03 abr. 2006.

BRAGA, Newton. Módulos inteligentes LCD multi-matrix. **Saber Eletrônica**, São Paulo, ano 17, n. 201, p. 11-23, set. 1989.

CHECK control tem mais utilidade do que um computador de bordo. **JC Online**, Recife, 14 mar. 1999. Editoria Veículos. Disponível em: <http://www2.uol.com.br/JC/_1999/1903/vc1403b.htm>. Acesso em: 02 abr. 2006.

FREESE, Cristiano. **Protótipo de um computador de bordo automotivo baseado em PC Linux**. 2003. 134 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau, Blumenau.

FUTURE ELETRONICS. **Future eletronics**: NXP semiconductors. [S.l.], 2006. Disponível em: <http://www.futureelectronics.com/promos/philips>. Acesso em: 02 out. 2006.

LABCENTER ELETRONICS. **Labcenter eletronics**. [S.l.], 2006. Disponível em: <http://www.labcenter.co.uk/>. Acesso em: 01 out. 2006.

MICROCONTROLADOR. In: WIKIPÉDIA, a enciclopédia livre. [S.l.]: Wikimedia Foundation, 2006. Disponível em: <http://pt.wikipedia.org/wiki/microcontrolador>. Acesso em: 26 abr. 2006.

OFICINA BRASIL. **Teste do sensor de temperatura do líquido de arrefecimento**. [S.l.], 2006. Disponível em: <http://www.oficinabrasil.com.br/home/ler.news.asp?AreaBanner=3&codItem=1607>. Acesso em: 30 maio 2006.

PHILIPS ELETRONICS. **Philips semiconductors**. [S.l.], 2006. Disponível em: <http://www.semiconductors.philips.com>. Acesso em: 02 abr. 2006.

PIROPO, Benedito. **Fórum PCs**. [S.l.], 2006. Disponível em: <http://www.bpiropo.com.br/fpc20050124.htm>. Acesso em: 15 maio 2006.

RISNIK, David M. **Velocímetro digital**: CCS-28. São Paulo, 2006a. Disponível em: <http://www.geocities.com/SiliconValley/Program/3430/velocimetro.htm>. Acesso em: 02 abr. 2006.

RISNIK, David M. **Conta giros**: SYS3. São Paulo, 2006b. Disponível em: <http://www.geocities.com/SiliconValley/Program/3430/cg.htm>. Acesso em: 02 abr. 2006.

SUPERINTENDÊNCIA DA ZONA FRANCA DE MANAUS. **Sensores automotivos, contribuição para controle do carro**. Manaus, 2006. Disponível em: <http://www.suframa.gov.br/minapim/news/visArtigo.cfm?Ident=103&Lang=BR>. Acesso em: 30 maio 2006.

THOMAS, Martin. **ARM projects**. [S.l.], 2006. Disponível em: <http://gandalf.arubi.uni-kl.de/avr_projects/arm_projects/#winarm>. Acesso em: 01 out. 2006.

TREVOR, Martin. **The insirders guide to the Philips ARM7 based microcontrollers**. [S.l.]: Hitex, 2006. Disponível em: <ftp://hitex.podzone.net/pub/hitex/lpc2000/lpc-arm-book_srn.pdf>. Acesso em: 16 out. 2006.

ULRICH, Radig. **Elektronic, mikroelektronik, computer**. [S.l.], 2006. Disponível em: <http://www.ulrichradig.de>. Acesso em: 16 out. 2006.

WINSTAR DISPLAYS. **Welcome to Winstar.** Taipei, 2006. Disponível em: <http://www.winstar.com.tw>. Acesso em: 03 abr. 2006.