Divisão do trabalho:

**Introdução**: - Abordagem (A)

- Objetivos (A)

- Estado de Arte (A)

**Arquitetura**: - Abordagem (R)

- Descrição dos Blocos (R)

Hardware: - Material utilizado (microcontrolador Atmega 328P, baterias (4\*3.7V), regulador de tensão lm7805, motores DC 3 a 6 V, Ponte H L298N, SONAR HC-SR04, módulo Bluetooth HC-05, Servo motor SG90) (R)

- Esquema elétrico (numerar e associar partes ao diagrama de blocos) (R)

- PCB (R)

**Software**

- Fluxogramas e abordagens (A)

- Configurações (A)

**Resultados**

- Abordagem e edição do vídeo

**Conclusões** (A e R)- Abordagem

- Pontes fortes

-Limitações e futuras melhorias

- Considerações finais

**Referências** (A e R)

-

No âmbito da Unidade Curricular Laboratório de Sistemas, foi proposta a conceção de um projeto com o respetivo protótipo funcional que recrutasse conhecimentos adquiridos ao longo do curso. Neste sentido, comprometemos-mos no desenvolvimento de um veículo autónomo, explorando posteriormente o seu controlo remotamente.

O presente relatório expõe os aspetos mais pertinentes que foram tidos em consideração durante o desenvolvimento do protótipo, destacando as principais dificuldades e realçando os conhecimentos adquiridos.

O compromisso para o desenvolvimento de um AGV passou pela estipulação das metas a serem atingidas. Com foco sempre na elaboração de um projeto eficaz e também económico foram enumerados os seguintes objetivos:

O desenvolvimento de todo o código foi realizado em linguagem C no ambiente de desenvolvimento Atmel Studio 7 conforme requerido, onde se torna notório a simplicidade na programação do ATmega328P frente à linguagem Assembly

Antes de iniciar a programação foi necessário averiguar qual seria a configuração dos fuse bits, responsáveis por definir a velocidade e tensão pela qual o microcontrolador irá operar. Tratam-se de 3 bytes de memória não volátil que podem ser reconfigurados sempre que necessário. A sua leitura pode ser feita usando o software AVRDUDESS que também permite a configuração destes mesmos bytes. Auxiliarmente usou-se a calculadora online [Engbedded Atmel AVR® Fuse Calculator](https://www.engbedded.com/fusecalc/) paralelamente com as informações prestadas pelo [datasheet](https://datasheetspdf.com/pdf-file/1057332/ATMEL/ATmega328P/1) do microcontrolador.

Verificou-se que o ATmega328P operava a 1MHz por defeito a partir de circuito oscilador RC interno com Fuse Low Byte a 0x62, Fuse High Byte a 0xD9, Extended Action a 0xFF e LockBit a 0xFF.

Para o desenvolvimento do c&oacute;digo capaz de instruir instruções a serem efetuadas pelo AVR em questão foi necessário recorrer a um conjunto de bibliotecas. Segue uma breve descrição das mesmas.

* <avr/io.h> - Permite a utilização dos I/O do microcontrolador e contém os nomes da maioria dos seus registos.
* <avr/interrupt.h> - Permite a utilização das interrupções do microcontrolador com maior comodidade através dos macros que esta estabelece, como por exemplo a ativar/desativar globalmente as interrupções (bit I do SREG).
* <util/delay.h> - Permite o estabelecimento de um atraso propositado, como por exemplo a necessidade de um intervalo de espera enquanto o servo motor finaliza uma rotação (delay mecânico). Aconselha-se o estabelecimento da frequência a que o microcontrolador está a operar segundo o um define (#define *F\_CPU* 1000000UL).

Com o intuito de tornar o código mais legível foram estabelecidos vários defines devidamente interpretados pelo compilador. Segue uma breve descrição dos mesmos.

* #define \_500ms 244 - Valor que permite de certa forma aumentar intervalo de tempo estabelecido para a comutação do estado do LED (que opera a 1 Hz) durante a interrupção de overflow do timer0
* #define pwm(val) 255\*val/100 - Expressão responsável por converter o valor introduzido em “val” (duty cicle) percentual para uma base binária (0 a 255)
* #define dist\_minf 50 - Distância mínima estabelecida para a zona de perigo do sonar da frente
* #define dist\_mint 30 - Distância mínima estabelecida para a zona de perigo do sonar de trás

As configurações iniciais dos registos do microcontrolador permitem uma correta utilização dos periféricos deste além de estabeleceram qual o funcionamento das propriedades de que este é constituído.

Deste modo, foram utilizadas duas interrupções externas (para a leitura do estado do pino ECHO de dois módulos ultrassónicos) onde a deteção do flanco ascendente ou descendente é conseguida dado a alteração cíclica do registo EICRA a cada passagem pela rotina de interrupção e pela função responsável por ativar a operação dos sonares.

Relativamente aos timers, procurou-se a otimização utilizando o TIMER0 em modo Fast PWM com ambas as saídas OC0A e OC0B (output compare) ativas responsável por gerar uma interrupção de overflow a cada 2,048 ms passados. Este tempo auxiliado com o estabelecimento da constante \_500ms permitiu a alteração do estado o LED de 500 em 500 ms. Adicionalmente foi possível obter dois sinais PWM para fins de controlo da velocidade de rotação dos motores conforme anteriormente explicado alterando o valor dos registos OCR0A OCR0B. Seguem-se os cálculos necessários. (ATENÇÃO EXPLICAR ATRÁS).

Já o TIMER1 foi também configurado em modo Fast PWM, mas apenas com OC1A ativo para fins de controlo da posição do servo motor. A utilização deste timer de 16 bits no modo 14 permitiu gerar um uma onda quadrada com frequência de 50 Hz necessária para o correto funcionamento do servo motor. Só neste modo foi possível obter tão elevada precisão. A onda quadrada foi conseguida fixando o valor 19999 no registo ICR1. A variação do duty cycle da onda foi possível através do registo OCR1A. Seguem-se os cálculos necessários.

Por fim, o TIMER2 também configurado em modo Fast PWM, responsável por gerar uma interrupção por overflow a cada 16,384 ms. Este timer configurado neste modo possui o tempo suficiente para fazer o cálculo da distância dos sonares traseiros sem ser necessário a implementação de um contador por software como observado no TIMER0, conforme será explicado à frente. Seguem-se os cálculos necessários.

Relativamente às configurações da USART para estabelecer uma comunicação wireless Bluetooth com recurso ao módulo HC-05 foi alocado o valor 12 no registo UBRR0, conforme o recomendado no datasheet do ATmega328P a uma frequência de 1 MHz para um baud rate de 9600bps, para uma operação ao dobro da velocidade.

As restantes configurações podem ser consultadas na figura X com a devida justificação relativamente à sua implementação.

De forma a expor a solução implementada segue-se análise baseada em fluxogramas e, caso seja necessário, a apresentação de alguns trechos de código.

Na função Main são feitas as inicializações executando a função init() e colocando o servo motor na posição central (0º). Antes de entrar no ciclo infinito é enviada uma mensagem ao utilizador “AGV READY……”. Neste momento é possível ao utilizador interagir com o AGV através do botão ON/OFF presente na interface da aplicação utilizada – *Bluetooth Electronics* – figura x+1. Botão ON inicia o modo controlado, caso contrário inica o modo automático.

O módulo automático executa as instruções recebidas da última comunicação série USART. Conforme é possível observar, a interface com o utilizador é constituída por um terminal que apresentas as mensagens envias e recebidas. Ainda a baixo, uma caixa de texto onde podemos enviar mensagens personalizadas. O controlo no canto inferior direito permite deslocar o AGV nas oito direções apresentadas. Por fim, no canto inferior esquerdo, uma barra deslizante capaz de controlar a velocidade do veículo (estava velocidade encontra-se limitada a 6V via software, correspondente a 104 numa escala de 0 a 255).

Na ocorrência do AGV operar em modo automático é verificado se o mesmo se encontra numa fase rotatória provocada deteção de um obstáculo. Neste cenário o sonar é colocado na posição central e o protótipo encontra-se a rodar (direita ou esquerda) seguido de uma série de leituras da distância frontal a qualquer objeto. Quando esta distância for maior que 80 cm (distância de segurança da frente) o AGV para de rodar e potencialmente inicia a progressão em frente. Se este cenário não for verificado é feita uma deteção do estado do veículo.

O estado 0 indica que o carro está a deslocar-se em frente e a fazer um varrimento constante numa amplitude de -20º a +20º do espaço à sua frente (esta pequena rotação permite a deteção de obstáculos com fornas irregulares, não paralelas ao sonar). A deteção de um obstáculo faz com que o AGV se desloque para trás durante um segundo ou um dos sonares traseiros detetarem um obstáculo a menos de 30 cm (distância que define a zona de perigo). É alterado o estado do AGV para 1 e guardada a posição do servo motor no momento da deteção do objeto para futura reposição desta.

Neste momento o veículo encontra-se parado e em condições de executar a condição de estado 1. O servo motor coloca o sonar virado para a esquerda e aguarda a sua estabilização. Por fim faz a leitura da distância lida pelo sonar nesta posição para futura comparação.

É então iniciado o estado 2, pelo que o servo motor coloca o sonar virado para a direita. Após as mesmas restrições da espera necessário para a rotação do sonar é feita a leitura da distância lida pelo sonar e comparada com o valor anteriormente guardado. Na situação destas serem iguais o AGV opta por rodar para a direita. É então reposta a posição do servo motor, redefinido o estado a 0 e iniciada a rotação do veículo. O ciclo é repetido até mudança do modo de operação de automático para controlado.

Ao longo do seu funcionamento o servo motor é responsável por coloca o sonar em cinco posições diferentes. Imaginando uma semicircunferência em que conseguimos definir uma rotação de -90º a +90º e sendo a sua orientação central alocada a 0º podemos estabelecer a seguinte tabela

O sistema implementado procura a otimização de recursos, desta forma foi desenvolvido um algoritmo que permita alocar a responsabilidade de controlo da velocidade de rotação dos motores, comutação do estado do LED sinalizador de sistema ativo à frequência de 1 Hz e contagem dos tempos necessários às leituras do sonar frontal. O modo implementado permite entrar na rotina de overflow presente na figura x+2 a cada 2,048 ms, processando-se os seguintes acontecimentos:

Controlo do LED sinalizador de sistema ativo – Operação responsável por comutar o estado do LED a cada 500 ms. É realizado um decremento da variável cont\_timer iniciada com o valor do define \_500ms. Assim que este contador alcançar o valor zero, este é reiniciado e é alterado o estado do LED.

Incremento do contador associado ao módulo ultrassónico frontal – Na ocorrência deste módulo se encontrar em espera pelo fim do impulso de ECHO por ele gerado, esta condição fica responsável por contar o número de overflows ocorridos que seguidamente serão processados noutra função.

Controlo da velocidade de rotação dos motores – Segunda as configurações deste timer e tendo as saídas OC0A e OC0B ativas é possível alterar o duty cycle das ondas quadradas responsáveis por controlar a velocidade de rotação de cada motor alterando apenas o valor de registo OCR0A e OCR0B ao longo do código.

A fim de contabilizar o tempo decorrido desde a deteção do estado alto do sinal ECHO após o envio do impulso sonoro por parte de um dos três sonares foram utilizadas as duas interrupções externas contidas no ATmega328P. Estas seguem um raciocínio idêntico, porém optamos pelo uso da interrupção externa zero em sincronia com o TIMER0 para a contagem de tempo do sonar frontal e a interrupção externa um em sincronia com o TIMER2 para a contagem de tempo dos sonares traseiros.

Ambas as interrupções são iniciadas com a deteção de uma subida de flanco no respetivo pino sendo seguidamente configurado o EICRA para que a próxima interrupção seja gerada por um flanco descendente.

Relativamente à interrupção externa zero, a deteção de uma subida de flanco origina a leitura do valor do TCNT0 (*Timer/Counter Register 0*) e inicialização das flags necessárias. Já na interrupção externa um há também uma inicialização de flags e o TIMER2 é ligado e reiniciado.

Segue-se a deteção da próxima ocorrência, flanco descendente do sinal ECHO. Relativamente à interrupção externa zero, é feita uma nova leitura do valor do TCNT0 seguido dos cálculos associados para a obtenção da distância medida e desbloqueado o valor da flag a 0 indicando a obtenção de uma nova distância frontal. A solução implementada para esta interrupção necessita de apenas de uma flag de 2 bits (flag = 0 indica obtenção de nova distância, flag = 1 indica que vai entrar a primeira vez na interrupção, flag = 2 indica que vai entrar a segunda vez na interrupção e está pronta a calcular a nova distância). No entanto, para a interrupção externa um é utilizada duas flags de 1 bit (f\_t = 1 indica primeira vez na interrupção e f\_t = 0 indica segunda vez na interrupção) e flag\_t (flag\_t = 0 aguarda processamento de nova distância e flag\_t = 1 indica obtenção de nova distância). Assim é ativada a 1 a flag\_t e feito o cálculo da nova distância traseira.

Com o intuito de apresentar o raciocínio acima explicado deixamos um trecho de código da rotina de interrupção INT0, idêntica à rotina INT1 alterando apenas as configurações do registo EICRA, flags utilizadas e cálculo, que de certa forma se tona mais simples, sendo apenas necessário dividir o tempo (em *us*) decorrido entre alteração do estado dos flancos do sinal ECHO por 58 para a obtenção de uma distância em *cm*.

Na figura x+3 é apresentado o fluxograma relativo a esta parte onde podemos observar de forma simplificada o algoritmo implementado relativo às interrupções externas zero e um.

Para os disparos dos módulos ultrassónicos foi implementada a função void triggersonar<span style='font-family:Consolas; font-size:13px;color:navy;'>()</span>, idêntica para os 3 sonares, responsável por colocar a 1 o pino onde está ligado o TRIGGER do respetivo módulo ultrassónico durante um período de pelo menos 10 us repondo depois o seu valor nulo. A vantagem de operar com o microcontrolador segundo uma frequência interna de 1 MHz é o facto de cada ciclo de clock levar cerca de 1 us. Com o auxílio da ferramenta <i>Processor Status</i> do ambiente de desenvolvimento Atmel Studio 7, durante a simulação foi possível verificar que decorrem cerca de 11 us fruto dos 6 comandos seguidos à ativação do TRIGGER do sonar. Considerando como exemplo o sonar frontal, o disparo deste fará atuar uma interrupção externa devido à ocorrência de um sinal com flanco positivo (INT0 conforme explicado anteriormente) a qual será responsável por obter uma nova distância lida. O processo termina quando a <span style='font-family:Consolas; font-size:13px;color:navy;'>flag</span> for 0 outra vez.

Seguindo o raciocínio que foi feito para as rotinas de interrupções externas, apresentamos um excerto de código ilustrativo da explicação acima enunciado com o respetivo fluxograma presente na figura x+4.

Uma última interrupção é gerada sempre que ocorra uma receção correta de informação por parte da USART. A ativação da flag RXC0 do registo UCSR0A informa-nos do sucedido. A informação pode ser lida do registo UDR0 neste momento. Igualmente pode ser envia informação de forma a obter algum feedback no momento da programação do AGV escrevendo o pretendido neste mesmo registo somente quando a flag UDRE0 estiver a 1.

As mensagens enviadas correspondem a um char (8 bits) conforme apresentado na secção das configurações. Este valor é armazenado, portanto na variável data. A receção de um ‘A’ corresponde à ativação do botão do canto superior esquerdo da interface apresentada na figura x+1 e origina o modo de controlo via Bluetooth. Contrariamente a receção de um ‘B’ corresponde à colocação no estado OFF desse mesmo botão significando à ativação do modo automático por parte do AGV. De outra forma, a receção de um ‘W’ indica a vontade de alterar a velocidade do veículo, sendo seguida de dois a quatro bytes, os primeiros bytes (apenas um na situação da velocidade a aplicar apenas conter um algarismo, dois bytes se a velocidade conter dois algarismos e três bytes caso contenha três algarismos) são devidamente processados terminando a trama com um ‘Y’. Um exemplo desta trama pode ser visto dentro da caixa de texto da figura x+1.

As restantes letras recebidas indicam a vontade de mudança de direção proveniente do utilizador. Assim que o controlador é solto, é enviado um ‘S’ colocando o AGV em posição de STOP. A informação é enviada do dispositivo Android para o módulo HC-05 sempre que ocorrem mudanças de valor da mesma.

Na figura x+5 é apresentado o fluxograma ilustrativo desta parte onde podemos observar de forma simplificada o algoritmo implementado relativo à interrupção da receção de informação da USART.

A função <span style='font-family:Consolas; font-size:13px;color:blue;'>void</span> <span style="font-size:9.5pt;font-family:Consolas;mso-fareast-font-family:&quot;Times New Roman&quot;;color:#880000">rotar\_sonar</span><span style='font-family:Consolas; font-size:13px;color:black;'>()</span> é responsável por processar a posição em que servo motor coloca o sonar e deslocá-la para a seguinte, ou seja, considerando o sistema apresentado na tabela 2, o servo motor deve rodar segundo a seguinte ordem -20º &rarr; 0º &rarr; +20º &rarr;0º &rarr; -20º e por aí adiante. Na função main é verificado se o sonar se encontra em rotação e feito uma espera até que este sistema estabilize para fazer as devidas leituras da distância frontal pretendida.

Na figura x+6 é apresentado o fluxograma ilustrativo desta parte onde podemos observar de forma simplificada o algoritmo implementado relativo à  função de rotação da posição do sonar frontal.