Sumário

Sumário

Sumário

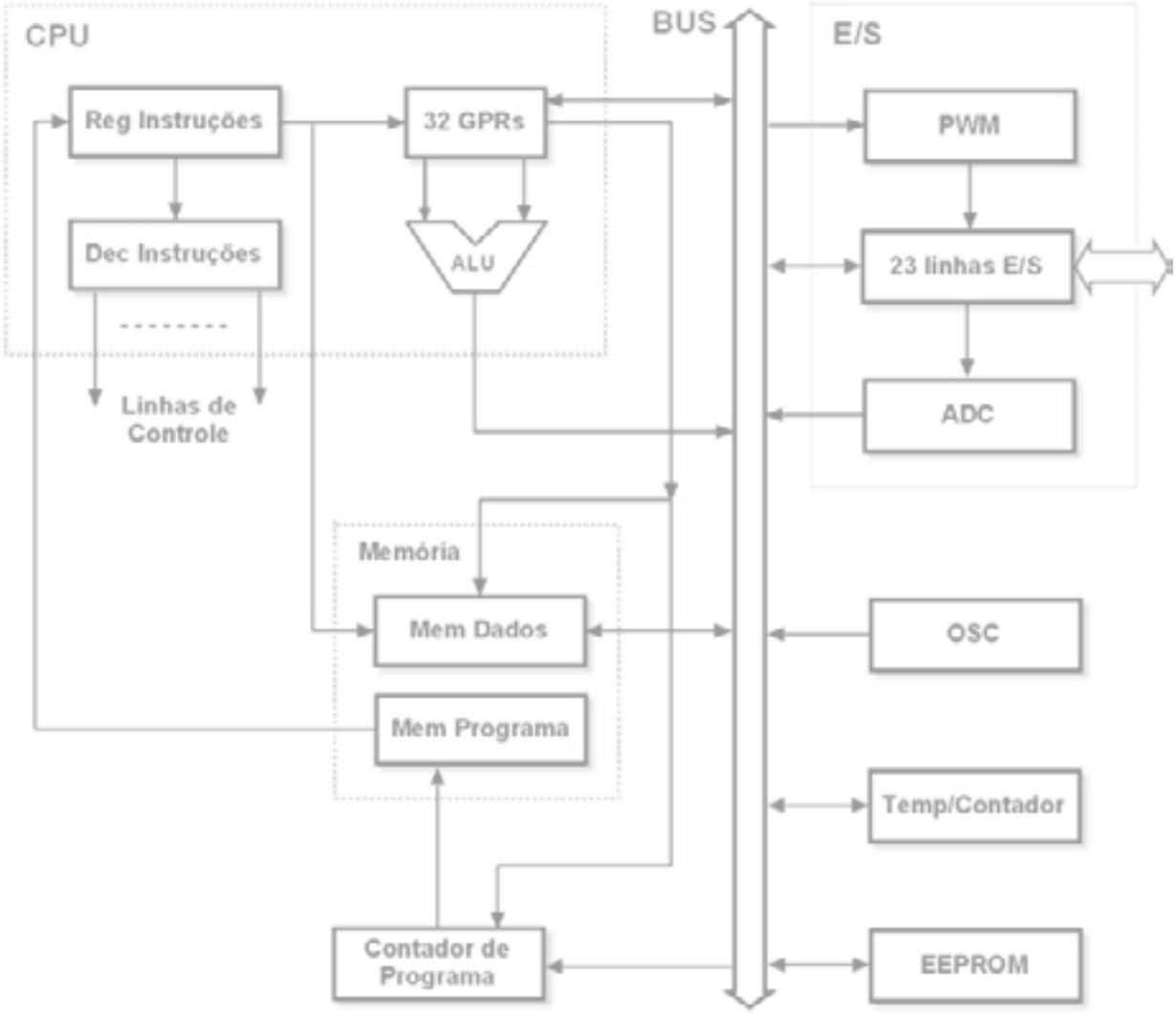
Sumário

Capítulo 1

Microcontroladores

Introdução
O microcontrolador AVR
A CPU e os registradores internos
Memórias e *clock*
Programação
Aplicações

Resumo do capítulo 1



Capítulo 1

Microcontroladores

Introdução

Até alguns anos atrás quando queríamos montar um circuito eletronico de média complexidade tínhamos que juntar muitos componentes, como resistores, capacitores, transistores, circuitos integrados, e até mesmo

indutores e chaves mecanicas de vários polos e posições. O circuito final era dedicado a uma só aplicação e não era muito fácil fazer correções ou melhoramentos uma vez montado, pois era preciso cortar filetes na placa de circuito impresso, refazer soldas e mesmo montar outros componentes sobre os já existentes. Os microcontroladores vieram para facilitar a vida dos projetistas e experimentadores de circuitos eletronicos mais complexos já que muitos dos fios e das trilhas metálicas e vários componentes discretos nas placas podem ser substituidos por linhas de código de programas que rodam nesses novos componentes. Linhas de programas são muito mais fáceis de mudar que componentes soldados numa placa de circuito impresso. Somente com a reescrita de umas poucas linhas do código de um programa podemos mudar radicalmente o funcionamento do nosso circuito e testar novas possibilidades no mesmo período de tempo que levaríamos para trocar alguns resistores numa placa. Para isso o técnico e o engenheiro eletronico precisam de um conhecimento mínimo de uma linguagem de programação dedicada a esses novos dispositivos.

O termo microcontrolador é usado para descrever um sistema minimo que inclui uma CPU, memória e circuitos de entrada e saída, tudo montado num único circuito integrado, que pode ter de 8 a até mais de 100 pinos. Alguns microcontroladores podem vir com contadores decimais internos, conversores analógico-digitais, comparadores de tensão e circuitos de comunicação serial, tudo embutido no mesmo encapsulamento. Tambem chamados de **microcomputador em um chip**, os microcontroladores podem ser de 8, 16 ou mesmo 32 bits, conforme o barramento que conecta cada circuito interno. Outro termo hoje muito comum é se referir aos microcontroladores como controladores embarcados, já que eles são montados dentro do aparelho ou instrumento que controlam. A diferença dos microcontroladores para os microprocessadores como os que integram nossos computadores pessoais é que estes ultimos são formados por somente uma CPU de 8, 16, 32 ou 64 bits e necessitam de memória externa e circuitos para controle de entrada e saída de dados para formar um sistema inteligente de processamento e controle.

Os microcontroladores funcionam seguindo uma lista de instruções armazenadas em forma de códigos binários em uma memória de programa interna. Nos computadores pessoais, ou PCs, esse programa é armazenado em um disco magnetico rígido, conhecidos por HDs (*Hard Disks*). Essas instruções são apanhadas uma a uma da memória, decodificadas por circuitos lógicos internos à CPU e então executadas. Uma sequencia de códigos pode por exemplo instruir o microcontrolador a fazer com que um de seus pinos de saída acione o circuito de um aquecedor se a temperatura lida por um sensor em um outro pino agora de entrada decrescer a certo valor. As instruções para um microcontrolador tradicionalmente são escritas na linguagem *Assembly*, hoje porem existe uma tendência a se usar cada vez mais a linguagem C e outras mais fáceis de aprender e programar, como BASIC e PASCAL.

Existem muitos fabricantes de microcontroladores e os mais conhecidos são os das empresas Microchip, Atmel, Intel e Freescale, entre outras. Cada fabricante tem sua própria arquitetura interna de registradores e capacidade de memória e sua própria lista de instruções; portanto um programa escrito para um dado microcontrolador não pode ser executado por um de outro fabricante. A arquitetura interna mais simples de um microcontrolador consiste em uma CPU, memória de programa, memória de dados e circuitos de controle de entrada e saída de dados, todos interligados por um barramento (*bus*) de 8 ou 16 bits, e tudo encapsulado em um só circuito integrado. A CPU (*Central Processing Unit*) ou Unidade Central de Processamento é a parte do microcontrolador responsável pelas operações matemáticas e pelas operações lógicas, como AND, OR e NOT. Nos microcontroladores a memória interna é separada em dois tipos conforme sua função: memória de programa e memória de dados. A primeira armazena as instruções escritas pelo programador e que dizem o que o sistema terá que fazer. A segunda armazena as informações que são trocadas de forma temporária entre a CPU e os diversos circuitos internos, como contadores e registradores.

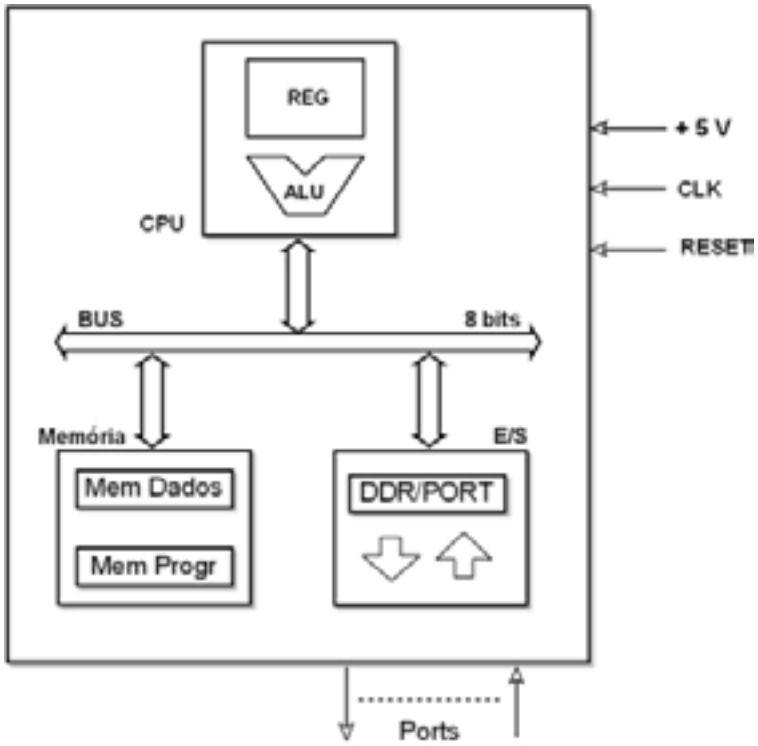


Figura 1: um AVR em blocos

Os microcontroladores normalmente são alimentados com uma tensão padrão de 5 volts. Muitos podem operar com tensões de 2 volts e outros com até 6 volts. Todos os microcontroladores necessitam de um circuito de relógio (*clock*) para sincronização de seus circuitos internos que normalmente são mantidos por um cristal ou ressonador cerâmico externo. A maioria dos pinos de um microcontrolador são usados para entrada e saída de dados, são os chamados *Ports*.

O microcontrolador AVR

A sigla AVR vem de *Advanced Virtual RISC* ou *Alf and Vegard RISC*, os nomes dos projetistas desse microcontrolador. *RISC* vem de *Reduced Instruction Set Computer*,

ou computador com um conjunto reduzido de instruções, que é uma referência ao pequeno número de instruções do microcontrolador se comparado aos primeiros microprocessadores cuja lista podia passar de 200 instruções.

O primeiro Arduino foi baseado no circuito básico com um microcontrolador AVR ATmega8 da empresa americana *Atmel Corporation* no ano de 2005 numa escola de

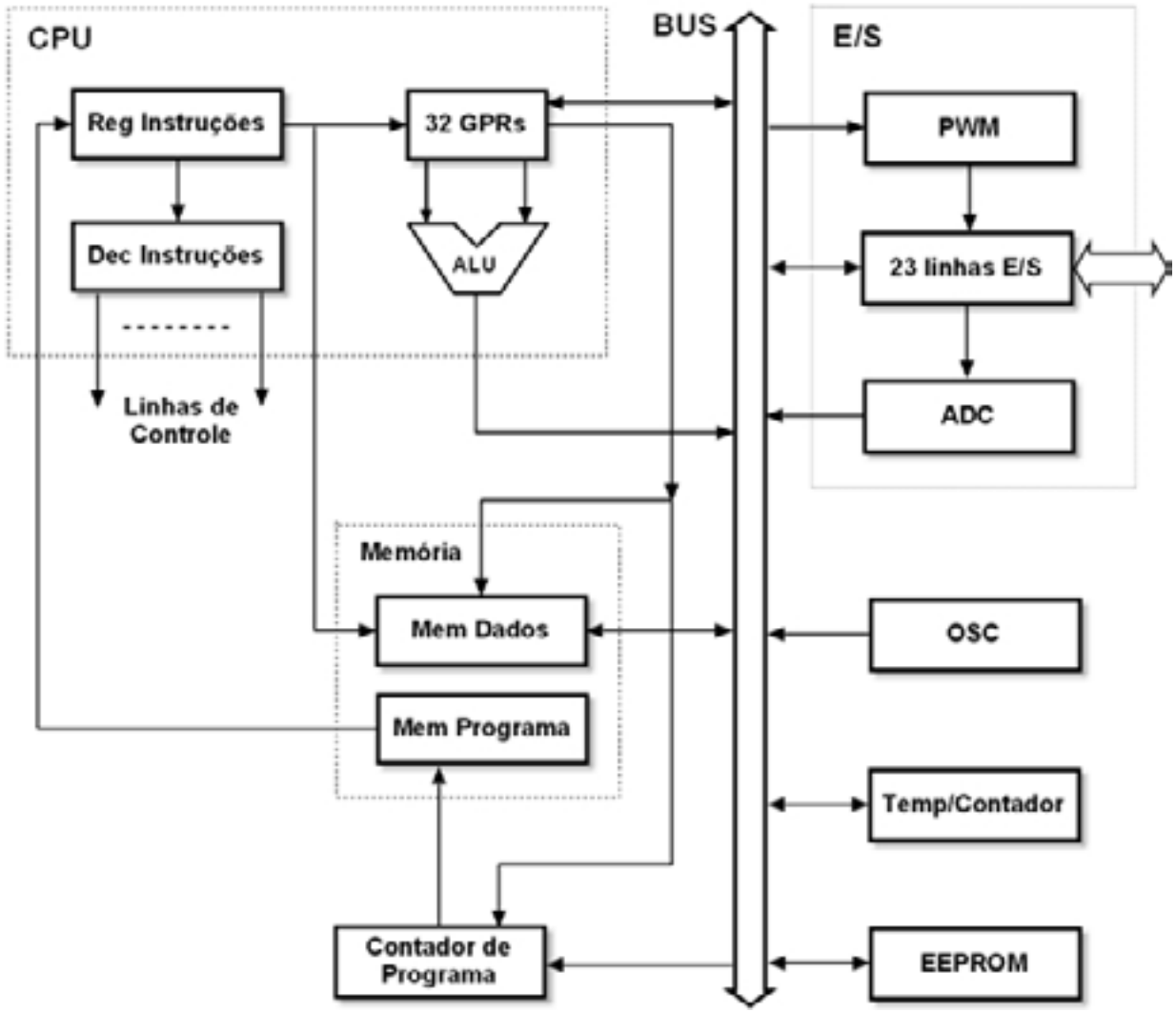


Figura 2: o microcontrolador ATmega 328 em blocos

artes interativas e *design*, na Italia. São tres blocos básicos: a CPU, o bloco de memória e o de registradores e circuitos de entrada e saída, veja o diagrama em blocos ao lado. A CPU é formada por uma ALU (*Arithmetic Logic Unit*) ou Unidade de Lógica e Aritmética e um conjunto de registradores de uso geral. O bloco de memórias agrega as memórias de programa e memória de dados. No bloco de E/S estão os *ports* que são circuitos de interfaces de entrada e saída, mais alguns registradores e

implementações de circuitos *pwm* e conversores A/D.
Depois de 2005 a Atmel lançou novas versões desse microcontrolador, com mais memória e outros recursos, porem mantendo a compatibilidade com aquele. Novos modelos de Arduinos surgiram com esses novos dispositivos. Os microcontroladores ATmega fazem parte do grupo mega da família AVR de microcontroladores de 8 bits da Atmel. Outro grupo dessa família são os microcontroladores **Tiny**, como o conhecido ATtiny13A de 8 pinos.

Na figura 2, acima, temos o diagrama simplificado em blocos do ATmega328. As tres principais blocos de qualquer microcontrolador ali estão reperesentadas: a CPU, a memória e o circuito de E/S. Na parte central à direita vemos o barramento de dados de 8 bits (1 *byte*) que interliga todos os módulos do microcontrolador que necessitam trocar informações. Normalmente os microcontroladores são projetados de acordo com uma arquitetura chamada *Harvard* onde os dados e os endereços são separados em dois barramentos independentes.

A CPU e os registradores internos

A CPU é formada pela ALU ou Unidade de Lógica e Aritmética, 32 registradores de uso geral, um Registrador de Instruções, um Contador de Programa (que aparece aqui fora da área tracejada), um Decodificador de Instruções e mais dois ou tres outros registradores especializados que no diagrama da figura 2 foram omitidos para maior clareza.

Um registrador é um bloco digital formado por um grupo de flip-flops do tipo D ou JK. No caso de um microcontrolador de 8 bits seus registradores podem ser blocos de 8 ou 16 flip-flops encadeados de modo que a passagem de bits de um flip-flop para outro possa ser controlada por linhas externas ao bloco. Cada flip-flop de um registrador pode ser carregado individualmente com um bit 1 ou 0, quando dizemos que o flip-flop foi setado ou resetado, respectivamente. Todo o registrador pode ser carregado com uma sequência qualquer de bits, ou ele pode ser todo resetado. Tambem podemos deslocar todos os bits em um registrador para a esquerda ou para a direita, uma ou mais posições. Todas essas operações dentro dos registradores podem realizadas por instruções dentro de programas ou por controle direto da CPU como resultado de outras operações internas.

Dentro da CPU do ATmega328 existe um grupo de 32 registradores de 8 bits conectados diretamente a ALU, a Unidade de Lógica e Aritmética. São os registradores GPR (*General Purpose Registers*) ou registradores de uso geral que são usados pela CPU para armazenar temporariamente os operandos, resultados das operações de soma

e subtração ou operações lógicas realizadas pela ALU. Aqui tambem são guardadas as variáveis locais e ponteiros de programas durante sua execução. O registrador de instruções (*Instruction Register*) armazena o *byte* de controle que representa uma instrução que recebe da memória de programa e o passa para o decodificador de instruções (*Instruction Decoder*), que é uma matriz de portas lógicas programáveis (*Programmable Array Logic*) cuja saída gera controles para as várias partes do microcontrolador.
O contador de programas (*Program Counter*) é um registrador que sempre aponta para o endereço na memória de programa da próxima instrução a ser executada. Ele é automaticamente incrementado depois que uma instrução é executada.

Memórias e clock

A memória do ATmega328 é composta por dois blocos principais: a memória de dados, que é do tipo volátil, RAM estática, para variáveis que não são guardadas em registradores e informações temporárias coletadas externamente; e a memória de programa que é do tipo não-volátil mas reprogramável, normalmente memória *Flash*, que guarda o programa que deve ser executado pelo microcontrolador. O ATmega328 possui 2K de memória para dados, 32K de memória para programa e tambem 1K de memória do tipo EEPROM, não-volátil e reprogramável, que no diagrama simplificado da figura 2 aparece fora do bloco de memórias, para armazenamento de informações que desejamos guardar mesmo depois de retirarmos a alimentação do circuito, como a senha em um *chip* no cartão de crédito.

Todos os microcontroladores requerem um circuito de relógio (*clock*) para sincronizar suas operações internas e externas. Esse sincronismo é realizado por um oscilador interno mas com uma referência externa, como um cristal e dois capacitores, um ressonador cerâmico ou mesmo um circuito RC. O ATmega328 pode operar com um cristal ou ressonador de até 20Mhz. Existem alguns blocos internos nos microcontroladores que trabalham sincronizados com o circuito de relógio, são os contadores e os temporizadores. Contadores são circuitos digitais montados com flip-flops, como os registradores, mas com a capacidade de incrementar ou decrementar um bit ou um pulso de cada vez, como o Contador de Programas visto acima. Contadores tambem podem ser resetados ou carregados com uma sequência de bits e a partir daí serem incrementados ou decrementados. Os contadores são tambem chamados de temporizadores (*Counters/Timers*) já que podem ser programados para indicar períodos decorridos a partir da contagem de um certo número de pulsos com duração fixa. O ATmega328 possui dois contadores de 8 bits e um de 16 bits de uso geral e um contador de tempo real.

O microcontrolador ATmega328 que estamos estudando é de encapsulamento em linha dupla (DIL ou *Dual-In-Line*) de 28 pinos, mais fáceis de se encontrar no mercado. Desses 28 pinos, 23 são pinos que podem ser programados para enviar (*output*) um nível lógico TTL (5 volts ou 0 volt) do microcontrolador para um circuito externo a ele conectado, ou podem ser programados para receber (*input*) um nível lógico TTL de um circuito externo para o microcontrolador. Esses 23 pinos são agrupados em dois conjuntos de 8 pinos e um de 7 pinos, que recebem o nome de *Ports*.

Nem todos os pinos nos microcontroladores são exclusivamente digitais, alguns deles podem receber sinais analógicos. No ATmega328 seis dos 23 pinos de entrada e saída podem ser programados para se tornarem entradas de sinais analógicos. Qualquer desses 6 pinos, após passar por um seletor analógico interno, é diretamente conectado a um conversor A/D (*Analog-Digital Converter*) de 10 bits de resolução. A tensão analógica na entrada desse conversor pode variar de 0 a +5 volts, no máximo. Tambem outros seis no ATmega328 podem ser programados para a função PWM (*Pulse Width Modulated*) ou modulação por largura de pulso.
Um sinal PWM tem a forma de uma onda quadrada com frequencia fixa porem com a largura de seus pulsos diretamente proporcional à amplitude de um outro sinal chamado de modulante. Um sinal modulante pode variar a largura dos pulsos de um sinal PWM entre 0% e 100% da sua forma original. Outras funções tambem disponíveis nos pinos desses microcontroladores são um comparador analógico de tensão e um canal de comunicação serial atraves de uma USART (*Universal Synchronous-Asynchronous Receiver-Transmitter*) implementada internamente.

Programação

Já sabemos que um microcontrolador é um pequeno computador em um *chip*; porem não bastam a CPU, a memória e os circuitos de entrada e saída para ele funcionar, é necessário tambem um programa que instrua esse *hardware* o que tem que ser feito. O *hardware* é a parte física, visível, o corpo do computador; o programa, ou *software*, é a parte invisível ou alma deste. O programa fica residente em algum tipo de memória do computador. Nos computadores pessoais *PC* ou *Mac* o programa fica no disco rígido; nos microcontroladores ele fica na memória de programa interna do tipo *Flash*. A memória *Flash* é uma versão da memória EEPROM (*Electrically ErasableProgrammable ROM*) e onde são gravados o *bootloader*, que é um programa opcional de inicialização do microcontrolador, e posteriormente os programas do usuário.

O Arduino tem um *bootloader*, um pequeno programa residente numa parte da memória *Flash*, chamado *bootblock*. Logo após o *reset* do microcontrolador o *bootloader*, ou simplesmente *boot*, como é tambem chamado, se comunica via interface serial com o computador para receber o programa do usuário, e o grava na parte restante da memória *Flash* do microcontrolador e logo em seguida o executa. O *boot* é gravado com um programador externo e o programa do usuário é gravado pelo *boot*.

Programas são listas com instruções escritas numa linguagem que um computador entenda e as execute. A linguagem que o computador entende é a chamada linguagem de máquina, são blocos com longas sequências de 1´s e 0´s que a CPU lê, interpreta e distribui ordens para as suas várias partes internas, e destas controles para circuitos e dispositivos externos. Mas não escrevemos programas diretamente na linguagem das máquinas, os escrevemos em linguagens que nós humanos entendemos, para depois entregarmos a um outro programa, um compilador, que os converte para a linguagem dos computadores. Os programas para o Arduino são escritos na linguagem C. Para escrever nossos primeiros programas em C basta conhecermos umas poucas estruturas de controle e suas regras. Existe um aplicativo onde podemos editar, verificar ocorrência de erros e compilar nossos códigos em C para o Arduino, é o Ambiente de Desenvolvimento Integrado ou IDE (*Integrated Development Enviropment*) do Arduino, que veremos em detalhes no capítulo 3.

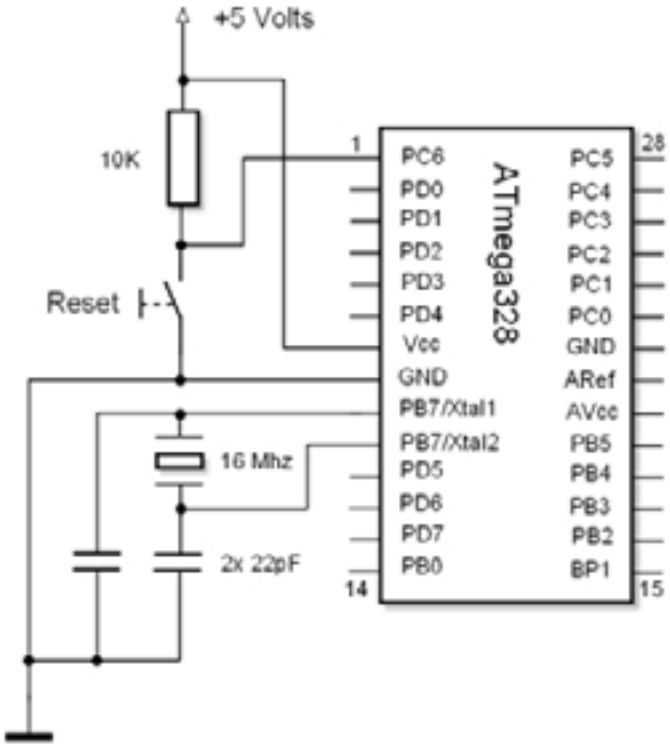


Figura 3: um sistema minimo com AVR

Aplicações

Um microcontrolador é uma ferramenta poderosa que permite ao projetista eletronico criar sistemas complexos sob controle de um programa. Encontramos microcontroladores em todos os aparelhos eletronicos que nos cercam, como

telefones celulares e GPS, televisores e seus controles remotos, fornos de micro-ondas e refrigeradores, impressoras e *mouses*, e até naqueles cartões de crédito e débito que possuem o já tão conhecido *chip* que carregamos em nossos bolsos. Um automóvel moderno pode ter até dez microcontroladores embarcados: dois ou tres para monitoração e controle do motor e transmissão, outro para os freios ABS, outro para os instrumentos no painel, para os alarmes e trancas das portas. O circuito mais simples que podemos montar com um microcontrolador ATmega328 precisa somente de um cristal de 16 Mhz, dois capacitores ceramicos de 22pF, um resistor de 10K e uma chave de *reset*, conforme vemos na figura 3, acima. para alimentá-lo qualquer fonte regulada que forneça 5 volts CC serve. Se nesse circuito simples conectarmos um sensor de temperatura no pino 23 (a linha 0 do *Port* C) e um transistor acionando um relé no pino 19 (linha 5 do *Port* B) , por exemplo; e na memória *Flash* do microcontrolador carregarmos um programa de umas poucas linhas poderemos controlar a temperatura interna de um forno elétrico mantendo-a entre dois limites previamente escolhidos. E se ainda conectarmos diretamente a outros pinos do microcontrolador um teclado simples e um mostrador LCD ou de 7-segmentos, bastaria incluirmos mais algumas linhas de código em nosso programa para entrarmos com os limites de temperatura desejados pelo teclado e termos a leitura instantanea da temperatura interna do forno no mostrador digital.

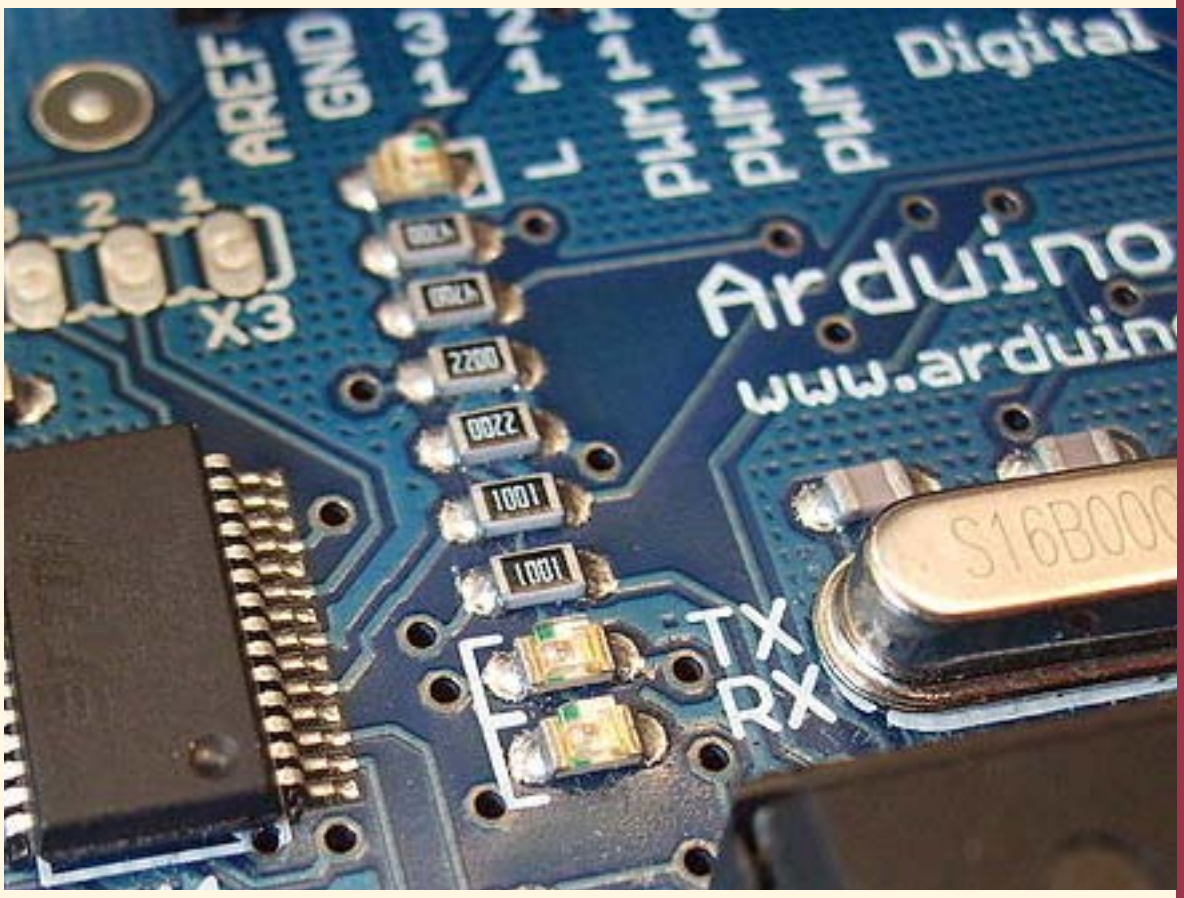
Resumo do capítulo 1

Um microcontrolador é um único circuito integrado que pode ter de 8 a até mais de 100 pinos e que inclui uma CPU, memória e circuitos de entrada e saída. Alguns modelos podem vir com contadores/temporizadores decimais, conversores A/D, comparadores de tensão e circuitos de comunicação serial, tudo embutido no mesmo encapsulamento. Os microcontroladores como qualquer computador funcionam seguindo uma lista de instruções armazenadas em forma de códigos binários em uma memória de programa. Essas instruções, são escritas em Assembly, C, BASIC ou Pascal, e apanhadas uma a uma da memória e decodificadas por circuitos lógicos internos à CPU e então executadas. A CPU é a parte do microcontrolador responsável pelas operações matemáticas e pelas operações lógicas; a memória é separada em dois tipos conforme sua função: memória de programa, do tipo *Flash*, e memória de dados, do tipo RAM; os *Ports* formam os circuitos de entrada e saída do microcontrolador.

O primeiro Arduino foi projetado com o microcontrolador ATmega8 da Atmel em 2005. Os mais novos Arduinos vêm com o ATmega1280 que tem 16 vezes mais memória de programa e tres vezes mais pinos de E/S. Registradores são blocos digitais formados por grupos de 8 ou 16 flip-flops do tipo D ou JK e são

usados pela CPU para guardar resultados de operações internas e troca de dados com circuitos externos. O ATmega328 possui um grupo de 32 registradores de uso geral de 8 bits conectados diretamente a ALU, a Unidade de Lógica e Aritmética. Outros registradores presentes em qualquer microcontrolador são: o Contador de Programas, o Registrador de Instruções e o Decodificador de Instruções. Cada grupo de *Ports* é tambem um registrador de 8 bits conectados fisicamente aos pinos do microcontrolador. No ATmega328 seis dos 23 pinos de entrada e saída que formam os *Ports* são diretamente conectados a um conversor A/D de 10 bits de resolução, enquanto outros seis pinos podem ser programados para a função de saída PWM, ou modulação por largura de pulso.

Hardware é a parte física, visível, que compõe o corpo do computador; o programa, ou *software*, é a parte invisível ou alma deste. Um microcontrolador sozinho é sómente uma combinação de circuitos eletronicos digitais montados em um mesmo encapsulamento que não tem nenhuma aplicação se não houver um *software* que instrua esse *hardware* no que tem que ser feito. E as instruções para o *hardware* são montadas em forma de listas escritas em uma linguagem inteligível para o programador, e depois convertidas para uma linguagem inteligível pela CPU do computador.

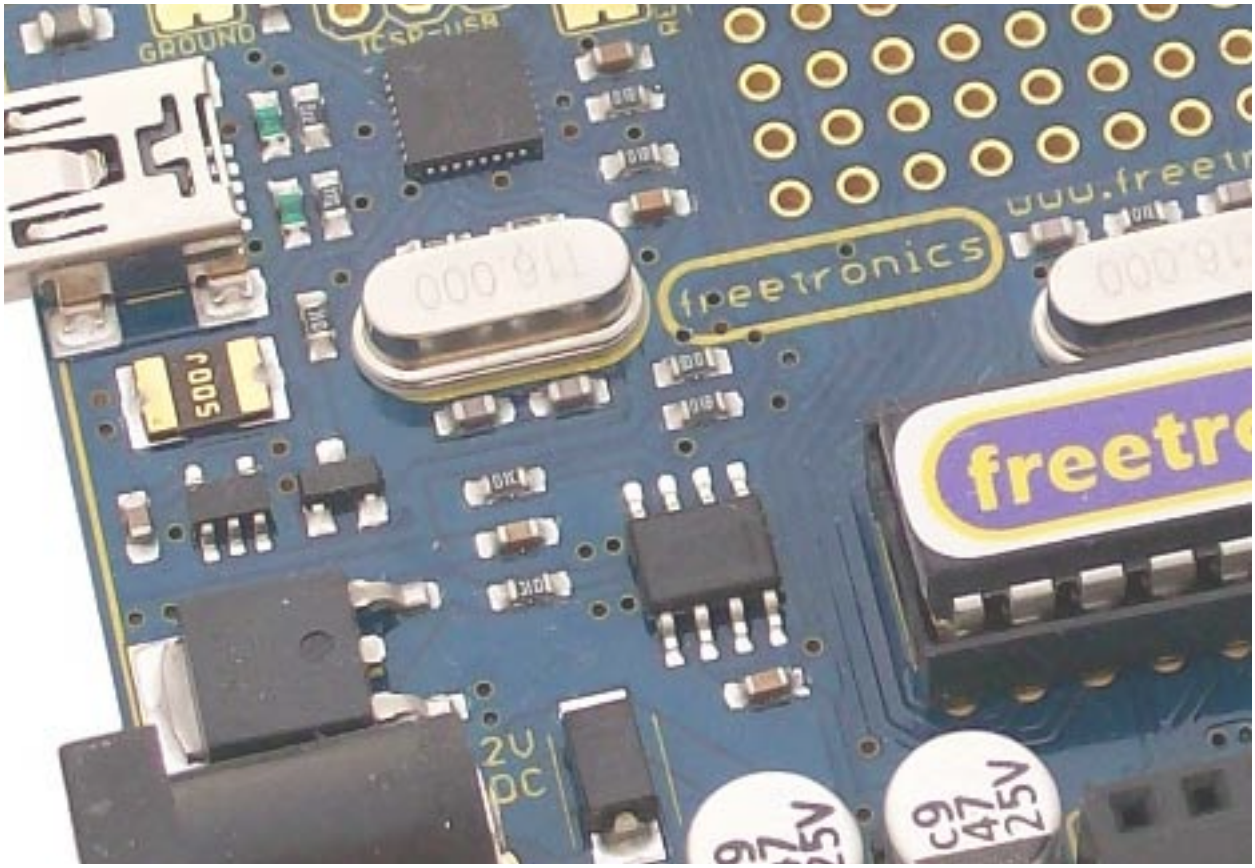


Capítulo 2

O *Hardware* do Arduino

Um pouco de historia
A placa do Arduino
Os *ports* digitais de E/S
As entradas analógicas e a saida *PWM*

Resumo do capítulo 2



Capítulo 2

O *hardware* do Arduino

Um pouco de história

O primeiro Arduino foi criado em janeiro de 2005 no Instituto de Interatividade e Design, uma escola de Artes visuais na cidade de Ivrea, Italia, a partir de uma ideia dos professores de Computação Física *David Cuartielles* e *Massimo Banzi*. O objetivo era criar uma ferramenta de *hardware* única que fosse facilmente programável por não especialistas em computadores e que tambem não fosse muito cara, para o desenvolvimento de estruturas interativas no curso de Arte e Design. Computação Física é o estudo da interação entre o nosso mundo físico e o mundo virtual dos computadores, e para isso envolve tambem o estudo de sensores eletricos e atuadores eletro-mecanicos.

À dupla de professores juntaram-se outros especialistas de programação que criaram um Ambiente de Desenvolvimento Integrado, uma ferramenta de *software* que traduz uma linguagem de alto nível como C para a linguagem de máquina que o Arduino entende. Tudo isso permitiu que artistas e designers pudessem dar às suas ideias uma forma física com protótipos que interagiam com os usuários e o ambiente onde estão.

Todo o projeto Arduino foi concebido segundo o princípio do *open source*, que em essência quer dizer que todos os seus programas são de domínio público, ou seja são livres para cópia, modificação e melhoramentos por qualquer pessoa. Da mesma forma os circuitos eletronicos tanto do Arduino quanto aqueles que foram criados para ser conectados a ele podem ser copiados e modificados. Existe na *web* centenas de comunidades de artistas, projetistas e programadores que disponibilizam suas criações para o mundo inteiro.

A placa do Arduino

O Arduino é composto por duas partes principais: um *hardware*, um conjunto básico de componentes eletronicos montados numa placa de circuito impresso, que é a plataforma para o desenvolvimento de protótipos; e um *software*, um aplicativo, o *bootloader*, residente na memória de programas do microcontrolador embarcado no Arduino. Existe tambem uma interface gráfica, um programa que roda num computador padrão PC em ambiente Windows ou Linux, ou ainda numa máquina Apple com o sistema operacional Mac OS X. É nessa interface gráfica ou Ambiente de Desenvolvimento Integrado (IDE – *Integrated Development Environment*) que o leitor vai criar seus programas e depois carregar para o *hardware* do Arduino. São esses programas, ou na linguagem do Arduino: *sketches*, que vão dizer ao *hardware* o que deve ser feito.

ARDUINO	Diecimila	Duemilanove168	Duemilanove328	Mega
Processador	ATmega8	ATmega168	ATmega328	ATmega1280
Memória flash	8 K	16 K	32 K	128 K
Memória RAM	1 K	1 K	2 K	8 K
Memória EEPROM	512 bytes	512 bytes	1 K	4 K
Pinos digitais	14	14	14	54
Pinos analógicos	6	6	6	16
Saídas PWM	3	6	6	14

O *hardware* do Arduino é baseado nos microcontroladores AVR da Atmel, principalmente nos modelos ATmega8, ATmega168, ATmega328 e no ATmega1280.

Conforme o microcontrolador utilizado o Arduino recebe um codinome em italiano. Veja na tabela comparativa abaixo as diferenças principais entre os Arduinos em relação ao microcontrolador nele embarcado.

O primeiro Arduino, batizado depois de Diecimila que em italiano quer dizer dez mil, marca de vendas desse modelo, usava o ATmega8 com uma fonte de alimentação simples com o regulador **LM7805**, um circuito de conversão para comunicação serial RS-232, e alguns conectores para os *Ports* do microcontrolador. Tambem foi acrescido ao projeto original um conector para a programação do ATmega8 no circuito, o *ICSP* ou *In-Circuit Serial Programming*. Veja o diagrama em blocos do Arduino na figura 4.

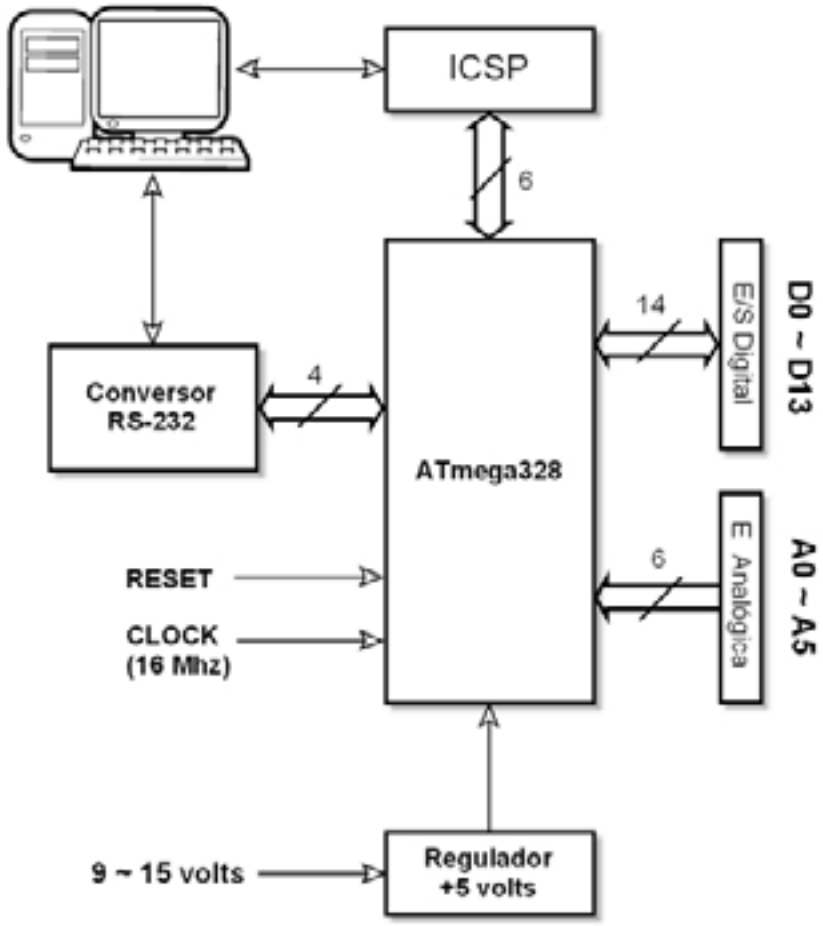


Figura 4: Diagrama em bloco do Arduino

O circuito do conversor RS-232 originalmente foi projetado com transistores bipolares **BC557**. Depois surgiram versões como o Arduino Freeduino MaxSerial que utilizavam o circuito integrado conversor **MAX232N**. A função do conversor RS-232 é apenas compatibilizar os níveis TTL do ATmega8 com os níveis de tensão padronizados do protocolo RS-232 na comunicação com um computador PC ou Apple.

Na placa do Arduino o conector macho ICSP (*In-Circuit Serial programming*) de 6 pinos é ligado diretamente a alguns pinos do microcontrolador e é conectado atraves de um cabo com a porta paralela ou serial de um computador, para a programação do *bootloader* no microcontrolador. A fonte de alimentação de 5 volts é composta por um único regulador integrado LM7805 e filtros capacitivos. A entrada da fonte pode receber tensões de 9 a 15 volts não regulados de um carregador de baterias ou de uma fonte de bancada. Tambem é possível alimentar o módulo com uma bateria comercial de 9 volts.

Por fim o Arduino tem dois outros conjuntos de conectores: um com 14 pinos para entradas ou saídas de sinais digitais e um de 6 pinos para entrada de sinais analógicos. Veremos mais adiante que esses pinos de entradas analógicas tambem podem ser programados para entradas ou saidas digitais, totalizando 20 pinos para esse fim. No módulo temos ainda um botão para resetar o microcontrolador e um LED que vai conectado ao pino digital 13 para testes simples e monitoração.

Figura 5: A placa do Arduino modelo Duemilanove

O Arduino possui um total de sete conectores. Veja a figura 5 acima. São dois conectores com os 14 pinos digitais que ficam na parte superior da placa e são identificados com serigrafia como "DIGITAL" e numerados de 0 a 13, da direita para a esquerda. Os pinos 0 e 1 são os dois pinos RX e TX de comunicação serial entre o Arduino e o computador. Na parte inferior da placa à direita fica o conector de 6

pinos identificado como “ANALOG IN” para sinais analógicos (tensões de entradade 0 a +5 volts). Aqui os pinos são contados da esquerda para a direita e com serigrafia são identificados com os números de 0 a 5. Esses pinos podem ser programados tambem como entradas ou saídas digitais da mesma forma que os 14 pinos digitais, podendo assim obtermos 20 pinos digitais. O pino de entrada de tensão de referência (“AREF”) para esse conversor fica no conector digital da esquerda, é o último pino. À direita deste pino fica um pino de terra (“GND”).

À esquerda de “ANALOG IN” existe um conector auxiliar identificado como “POWER” com tensões para alimentar um circuito externo eventualmente conectado ao Arduino. Um último conector ao lado botão RESET fica o ICSP. Uma vez programado o *bootloader* toda a programação do Arduino passa a ser feita pela porta serial USB que fica à esquerda da placa. Ainda à esquerda da placa, logo abaixo do conector serial, fica a entrada para a alimentação do Arduino, um conector tipo *jack* que pode receber tensões de 9 a 15 volts CC com o positivo no pino central.

Os Ports digitais de E/S

O termo *Port* se refere a um grupo de 8 pinos de um microcontrolador que são fisicamente conectados a registradores de 8 bits no circuito digital de E/S. O ATmega328 tem tres *Ports* que são conectados a tres registradores chamados de PORT B, com 8 flip-flops, PORT C, com 7 flip-flops, e PORT D, tambem com 8 flip-flops. Não existe o PORT A. Podemos enviar ou receber qualquer combinação de níveis lógicos por esses *Ports* físicos para esses registradores na forma de 1 byte. Tambem podemos programar individualmente cada pino de qualquer desses *Ports* de forma que uns possam ser saídas enquanto outros possam ser entradas de níveis lógicos. É importante conhecermos mais detalhadamente o circuito dos *Ports* porque são eles são a conexão física entre a CPU e o mundo exterior. Na figura 6 podemos ver o diagrama em blocos do circuito interno de um *Port* típico.

No ATmega328 cada *Port* tem na verdade dois registradores de 8 bits: um registrador de direção de dados chamado de DDR (*Data Direction Register*) e um registrador de dados propriamente dito ou simplesmente registrador PORT. O equivalente ao DDR nos microcontroladores PIC é o registrador TRIS. A saída Q de cada flip-flop do registrador DDR é que determina o sentido do fluxo de dados em cada pino do microcontrolador, se da CPU para o pino ou se do pino para a CPU. O registrador PORT somente lê e armazena o nível lógico que deve ser transferido do barramento para o pino. Quem comanda todas as linhas de controle tanto no DDR quanto no PORT é o Decodificador de Instruções que vimos no capítulo 1. O nível lógico nas

entradas D desses registradores são transferidos para suas saídas Q com o pulso de habilitação em suas linhas de *clock* (CLK).

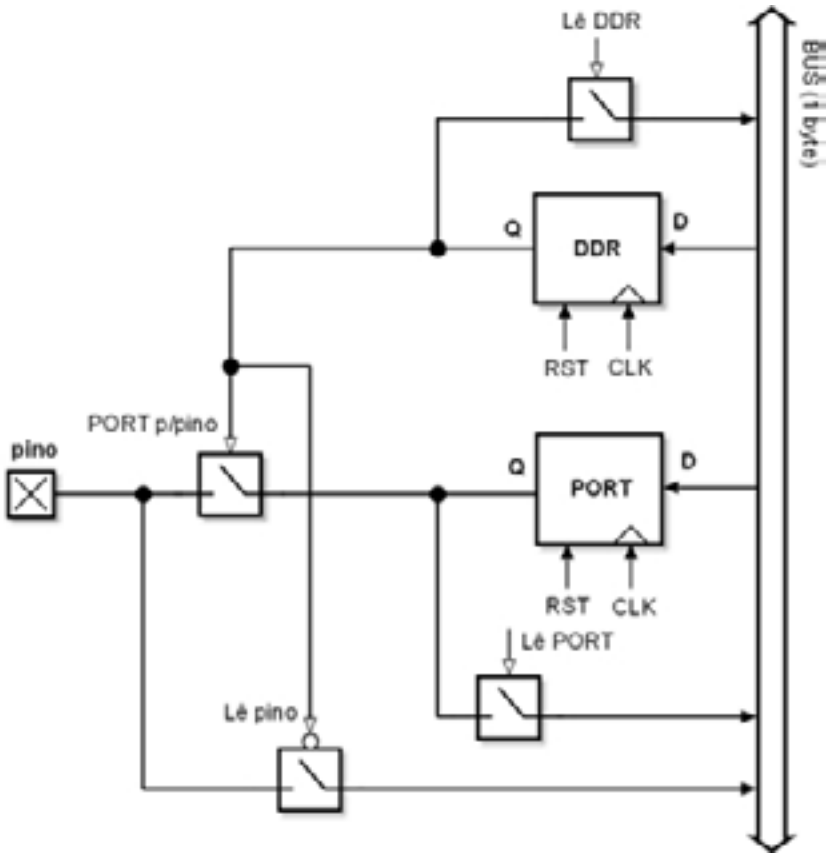


Figura 6: um Port de um microcontrolador ATmega328

O registrador DDR funciona da seguinte forma: se o nível lógico transferido de uma de suas entradas D para a correspondente saída Q for baixo (estado lógico 0) a chave digital nessa saída que vai ao pino a ela associada é aberta; com isso esse pino permanece no modo entrada, ficando em alta impedancia; a chave digital mais abaixo é fechada permitindo a transferencia de níveis lógicos externos para o barramento e daí para a CPU. Se o nível lógico na saída Q desse registrador DDR for alto (nível lógico 1) a primeira chave é fechada, o que faz com que a saída Q do registrador PORT seja transferida para o pino a ela associado, o pino permanece no modo saída. A segunda chave digital é aberta.

Nos microcontroladores *PIC* ocorre o contrário: o nível lógico 1 na saída do registrador de direção *TRIS* faz o pino de um *Port* ser entrada, o nível 0 faz esse pino ser saída.

Chaves digitais são portas não inversoras que possuem uma linha de controle para

ligar e desligar sua entrada de sua saída, são conhecidas como *tri-state buffers*. A qualquer momento as saídas Q dos registradores DDR e PORT podem ser lidas pela CPU atraves do controle de *clock* de outras duas chaves digitais. Esses registradores são resetados quando o microcontrolador é inicializado ou resetado e todos os pinos são colocados em estado de alta impedancia.

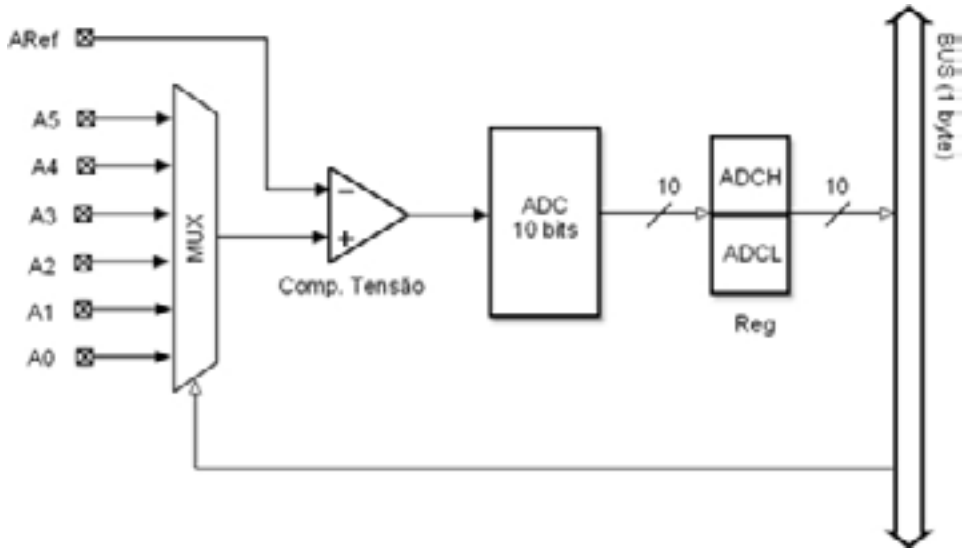


Figura 7: o conversor A/D do microcontrolador ATmega328

As entradas analógicas e a saída PWM

Seis dos pinos digitais do *Port* C do ATmega328 podem ser configurados para receber qualquer nível de tensão entre 0 e 5 volts que vai para a entrada de um conversor analógico-digital (A/D) de 10 bits de resolução. Somente um desses pinos pode ser selecionado de cada vez por um multiplexador (MUX), um tipo de chave seletora, para ser entrada do conversor. Os 10 bits que representam aquele nível de tensão contínua na entrada do conversor são armazenados em dois registradores internos, um de 8 bits e outro de 2 bits. Veja o diagrama em blocos na figura 7. Observe que entre o conversor A/D e o MUX existe um comparador de tensão que recebe uma referência externa de um dos pinos do ATmega328, o pino *Aref*, que determina a faixa de conversão para o circuito. No Arduino essas entradas analógicas são os pinos A0 a A5 (ou pinos 14 a 19).

Existe tambem no ATmega328 um circuito comparador de tensão independente que pode usado para monitorar dois níveis de tensão externos e disparar o contador digital interno *Timer/Counter1* quando uma dessas tensões for maior que a outra. As

entradas desse comparador são os pinos 6 e 7 do *Port* D. No Arduino são os pinos digitais 6 e 7.

Sistemas digitais tambem podem gerar sinais analógicos com circuitos conversores digitais-analógicos (D/A). Embora o microcontrolador AVR usado no Arduino não tenha um conversor D/A interno, é possível gerar voltagens analógicas de 0 a 5 volts em alguns de seus pinos de saídas digitais empregando uma técnica conhecida por PWM (*Pulse Width Modulation*) ou Modulação por Largura de Pulso. Com essa técnica pode-se construir um sistema eletronico que gera em sua saída um trem de pulsos TTL de cadencia fixa mas com a largura dos seus pulsos positivos variando de acordo com uma combinação de bits programada em um registrador de uso geral (GPR) do microcontrolador. Esse trem de pulsos PWM só precisa passar por um filtro RC passa-baixas semelhante àquele usado em fontes de alimentação para se obter uma tensão CC de 0 a 5 volts. No Arduino esses pinos PWM são os pinos digitais 3, 5, 6 e 9 a 11.

Resumo do capítulo 2

O primeiro Arduino foi criado em janeiro de 2005 na Italia por *David Cuartielles* e *Massimo Banzi*. O objetivo era criar uma plataforma de hardware que fosse facilmente programável por não especialistas em computadores. Todo o projeto Arduino foi concebido segundo o princípio do *open source*, onde todos os programas e circuitos são de domínio público.

O Arduino é composto por duas partes principais: um *hardware*, um microcontrolador e alguns componentes eletronicos montados numa placa de circuito impresso; e um *software* residente na memória de programas do microcontrolador e tambem uma interface gráfica que roda num computador padrão PC ou numa máquina Apple.

O *hardware* do Arduino é baseado nos microcontroladores AVR ATmega8, ATmega168, ATmega328 ou ATmega1280. Para se interligar ao mundo exterior o Arduino possui um conjunto de dois conectores: um com 14 pinos para entradas ou saídas de sinais digitais e um de 6 pinos para entrada de sinais analógicos.

Ports são a conexão física entre a CPU e o mundo exterior, são grupos de 8 pinos do microcontrolador que são fisicamente conectados a registradores de 8 bits no circuito digital de E/S e por onde podemos enviar ou receber qualquer combinação de níveis lógicos na forma de 1 byte. Cada *Port* tem dois registradores: o registrador de direção de dados DDR e o registrador de dados propriamente dito ou simplesmente PORT. O equivalente ao DDR nos microcontroladores PIC é o registrador TRIS.

Quem comanda todas as linhas de controle tanto no DDR quanto no PORT é o Decodificador de Instruções que fica na CPU do microcontrolador. Um nível lógico alto numa linha do DDR configura o pino correspondente como saída, um nível baixo como entrada. Nos microcontroladores PIC essa configuração é invertida.

O ATmega328 possui um conversor analógico-digital (A/D) de 10 bits de resolução no *Port* C com seis entradas que pode receber qualquer nível de tensão entre 0 e 5 volts. No Arduino essas entradas analógicas são os pinos A0 a A5 (ou pinos 14 a 19). Para gerar voltagens analógicas de 0 a 5 volts em alguns de seus pinos de saídas digitais o ATmega328 emprega uma técnica conhecida por PWM ou Modulação por Largura de Pulso. No Arduino esses pinos PWM são os pinos digitais 3, 5, 6 e 9 a 11.

No próximo capítulo vamos conhecer o Ambiente de Desenvolvimento Integrado, ou IDE (*Integrated Development Environment*), do Arduino, onde você leitor vai poder editar e depurar os programas, ou *sketches*, que serão gravados na memória *Flash* do ATmega328.