Microcontroladores:
Barramentos e Protocolos / UART

EMB5642 – Aula 10

UNIVERSIDADE FEDERAL
DE SANTA CATARINA

Introdução

- Nas aplicações com sistemas embarcados pode ser necessário realizar a comunicação entre o microcontrolador e um ou mais dispositivos externos. Estes dispositivos podem estar localizados tanto na mesma placa do circuito, como fora dela, a metros ou mesmo dezenas de metros de distância.

Categorias de comunicação

- As técnicas de comunicação podem ser divididas em duas grandes categorias: serial e paralela.
- Na comunicação serial, a informação a ser transmitida é fracionada em pequenas partes (bits) que são enviadas ao equipamento receptor uma após a outra, em série, daí a denominação de comunicação serial. Ex: interface seriais do PCs: RS232, USB, Firewire, protocolos de redes locais (Ethernet, Token-Ring). Outros protocolos: I2C, SPI, 1wire, LIN, CAN, etc.

Categorias de comunicação

- Na comunicação paralela, os bits componentes da informação são transmitidos simultaneamente (total ou parcialmente) em paralelo. Ex: Barramentos internos dos microprocessadores e microcontroladores, barramento ISA, PCI, VESA, AGP, a interface de impressora paralela dos microcomputadores, SCSI, IDE, etc.

Escolha...

- A escolha dentre os tipos de comunicação é um paradoxo da dualidade custo x benefício: na comunicação paralela, temos uma alta velocidade de comunicação, mas também uma utilização maior de meios de transmissão. Além de uma baixa imunidade a ruídos. Na comunicação serial, temos uma velocidade máxima menor que na comunicação paralela, mas também uma menor utilização de meios de transmissão e uma melhor imunidade a ruídos.

Protocolos de comunicação

- Nos protocolos síncronos, além das linhas de comunicação, encontramos uma ou mais linha de sincronização (clock). A informação na linha de dado é transmitida a cada transição na linha de clock. Neste tipo de protocolo, encontramos ainda uma classificação para os elementos de comunicação: o elemento que gera o sinal de sincronização de transmissão (clock) é chamado de mestre, e os dispositivos que recebem o sinal de sincronização são chamados de escravos.

Protocolos de comunicação

- Já nos protocolos assíncronos, não encontramos uma linha específica para sincronismo. Neste caso, a sincronização entre os elementos transmissor e receptor é garantida pela precisão dos clocks de cada elemento e também pela utilização de sinais marcadores de início (start) e fim (stop) da palavra transmitida.

USART – Universal Synchronous Asynchronous Receiver Transmitter

- Protocolo universal e possui dois modos distintos de trabalho.
  - Modo assíncrono: comunicação feita em duas vias, uma para transmissão (TX) e outra para recepção (RX), logo é full-duplex. Usada para implementar o padrão TIA/EIA 232. A sincronia é feita através da configuração do baud rate nos dispositivos e por bits de start e stop.
  - Modo síncrono: comunicação feita em duas vias, uma para clock (CK) e outra para dados (DT), half-duplex. Opera na arquitetura mestre-escravo, o bit menos significativo é enviado primeiro.

USART – Universal Synchronous Asynchronous Receiver Transmitter

- Os sinais da USART assíncrona pode ser alterados para o padrão TIA/EIA 485, que consiste em um par diferencial, half duplex ou dois pares diferenciais, full duplex, um par para RX e outro para TX. A vantagem é a robustez a interferência garantida pelo par diferencial.

SPI – Serial Peripheral Interface

- A comunicação SPI é síncrono e opera no modo full-duplex, suporta um dispositivo mestre e um ou mais escravos no mesmo barramento, uma característica é a simplicidade do barramento e mesmo transmissão dos dados, a velocidade máxima é de até 70 MHz.
- Atualmente está presente em inúmeros dispositivos, devido a capacidade de fluxo de dados e à facilidade de implementá-lo, por exemplo: microcontroladores, conversores DA e AD, sensores, SD Cards e outros.

SPI – Serial Peripheral Interface

- Clock – SCLK/SCK é utilizado para a sincronização entre o dispositivo mestre e o escravo. O protocolo especifica que o sinal deve ser simétrico.
- CS/SS – Chip Select é utilizado para selecionar ou habilitar o dispositivo com qual se deseja comunicar e também para encerrar a execução dos comandos transmitidos pelo dispositivo mestre.
- MISO/SDO – Serial data out é a linha utilizada para os escravos escreverem os dados a serem recebidos pelo mestre.
- MOSI/SDI – Serial data in é a linha utilizada para o mestre escrever os dados/comando a serem recebidos pelos escravos.

SPI – Serial Peripheral Interface

- Na comunicação SPI, os dispositivos escravos são selecionados por pinos CS (chip select), ao invés de possuírem um endereço e a comunicação é full duplex, linha separadas para escrita e leitura. Contudo não há padrão para o funcionamento do CS, com sua ativação podendo ser feita em nível lógico ‘1’ ou a mais comum em ‘0’.
- Os dados transferidos são normalmente representados por 8 bits, cuja transmissão pode ser iniciada pelo bit menos significativo (LSb) ou mais significativo (MSb), variando conforme o dispositivo.

I2C - Inter-integrated circuit

- O I2C é composto por duas linhas de comunicação, sendo uma linha de dados (SDA) e uma de clock (SCL). A linha SCL é controlada pelo dispositivo configurado como master, o qual é responsável pelo envio do sinal de clock para o dispositivo slave, enquanto a linha SDA é bidirecional (half-duplex) e carrega a informação.
- Esse barramento é do tipo multi-master, o que significa que mais de um dispositivo pode iniciar uma comunicação, desde que todos os outros estejam configurados como escravo.

I2C - Inter-integrated circuit

- O I2C permite que tenha-se 127 dispositivos no barramento (endereçamento de 7 bits) ou 1023 (endereçamento de 10 bits).
- Como os dispositivos trabalham no modo de dreno/coletor aberto nos pinos de comunicação I2C são necessários dois resistores de pull-up no barramento, uma para cada linha.
- As taxas de transferência são normalmente padronizadas, 100 Kb/s (standard mode), 400 kb/s (fast mode) e no máximo 3,4 Mb/s (high mode)

I2C - Inter-integrated circuit

- O endereçamento pode ser feito por 7 bits ou 10 bits, assim a comunicação é sempre iniciada com o endereço sendo posto na linha SDA pelo dispositivo mestre. O tamanho da informação que trafega na linha SDA é sempre em grupos de 8 bits, logo quando usa-se o endereçamento de 7 bits, o oitavo bit será o bit R/W qye define a direção do fluxo de dados. Se for um endereçamento de 10 bits serão necessários 2 bytes, que conterão 10 bits de endereço mais o bit de R/W.
- O barramento estará sempre ocioso enquanto as linhas SDA e SCL estiverem em nível lógico alto, assim a comunicação terá início quando houver a transição do nível lógico da linha SDA de alto para baixo enquanto a linha SCL estiver em nível lógico alto. Esta é a chamada condição de START.

I2C - Inter-integrated circuit

- Após a condição de START o sinal de clock é ativado pelo dispositivo mestre. Depois disso os bits vão sendo colocados um a um, na linha SDA, devendo o bit assumir o nível lógico correto enquanto o sinal SCL estiver em nível baixo, senão poderá ser considerado uma colisão de dados, resultando na transmissão sendo abortada.
- Ao final de cada byte transmitido o receptor tem que enviar um pulso ACK (acknowledge), pulso de reconhecimento, isso é realizado fazendo a linha SDA assumir nível lógico baixo no nono pulso de clock.

I2C - Inter-integrated circuit

- Para finalização da comunicação usa-se uma condição STOP, que ocorre quando há uma transição de subida em SDA enquanto SCL estiver em nível alto. Para informar ao dispositivo escravo que o mestre não deseja mais receber dados, o mestre envia um pulso NACK após o último byte recebido e impõe uma condição STOP no barramento.

CAN - Controller Area Network

- O barramento CAN foi desenvolvido para a comunicação de dados para a indústria automobilística. Com altos níveis de interferência eletromagnética, faixa grande de temperatura e umidade, este é um ambiente hostil para qualquer sinal e inclusive para dispositivos eletrônicos.
- O protocolo foi desenvolvido pela BOSH com a versão 2.0 em 1991 que foi aprovada pela ISO.

CAN - Controller Area Network

- Comunicação assíncrona, half duplex com velocidade máxima de 1 Mbit/s.
- A configuração é ponto a ponto, todos os nós são vistos como iguais. Porém há um mecanismo para prioridades, não se usa os termos mestre e escravo. RTR remore transmit request
- Valores lógicos no barramento são definidos como dominantes e recessivos.

CAN - Controller Area Network

- O acesso ao barramento é flexível, todos os nós podem iniciar uma mensagem. Um processo de arbitragem é aplicado em casos de acessos simultâneos, que não acarreta em perda de tempo nem dados.
- Há um número ilimitado de nós.
- Nós do barramento não possuem endereços, mas é aplicado um filtro de mensagem para determinar se os dados são relevantes para o dispositivo.

CAN - Controller Area Network

- Os dados são transferidos em frames, que possuem um formato complexo. O frame inicia com bits de identificação, que é o período de arbitragem, são 8 bytes permitidos por frame.
- Há um altíssimo nível de segurança de dados, como testes exaustivos para erros. Um nó problemático pode se desconectar do barramento.

USB – Universal Serial Bus

- O barramento serial universal é um barramento serial de par diferencial de baixa tensão que foi desenvolvido para conectar periféricos a plataformas computacionais. Trata-se de uma comunicação assíncrona padronizada, que opera no modo half-duplex, como sinal diferencial e codificação NRZI (Non Return to Zero Invert).
- O barramento é composto por 4 fios, um comum (GND), um Vcc(5V) e o par diferencial D- e D+ que carrega os dados.

USB – Universal Serial Bus

- Os dados são transmitidos usando uma codificação NRZI, e um sequencia especial de sincronia é enviada no início de todos os pacotes para que o dispositivo recebedor sincronize o clock, eliminando a necessidade de um fio para clock.
- O USB é plug-and-play, não há necessidade de desligar o PC para conectar/desconectar o dispositivo USB.

USB – Universal Serial Bus

- O USB provê uma alimentação de 5V que é disponibilizada para os periféricos, mas a corrente é relativamente baixa, de 100mA a 500mA, que razoável para periféricos como mouse, teclados ou pen drives, mas não para impressoras por exemplo.
- A grande vantagem dessa interface é a regulamentação dos dispositivos pela USB Implementers Forum que verifica e atesta a conformidade do produto com a especificação USB.

Topologia USB

- A topologia USB pode ser analisada por camadas, no topo há host (hospedeiro) com um root hub (hub raiz) integrado, que pode ser um PC ou dispositivo dotado de USB On-the-GO (OTG), que pode se comportar como host ou device, muito utilizado em tablets e smartphones.
- O host é o responsável pela inicialização dos serviços em um barramento USB e comunicação com um hub ou um ou mais dispositivos periféricos (que pode ser de até 127) através do root hub e podem haver no máximo 7 camadas com dispositivos e hubs.

Taxas de Transferência USB

- O barramento pode trabalhar em quatro taxas de transferência distintas:
- USB 1.0 – Low Speed 1,5 Mbps
- USB 1.1 – Full Speed 12 Mbps
- USB 2.0 – High Speed 480 Mbps
- USB 3.0 – Super Speed 5 Gbps
- Como a compatibilidade é a característica chave do USB, um dispositivo USB em uma versão anterior do padrão USB pode ser inserido numa plataforma de velocidade maior e operar na sua velocidade original.

Protocolo USB

- A maioria das transações consiste em três pacotes separados transmitidos no barramento, o controlador host inicia gerando um pacote token, este pacote indica o tipo e direção da transação, contém também o endereço dos dispositivo e end point. Endpoint é o ponto final do fluxo de dados entre o host e o dispositivo.

Protocolo USB

- Pode-se ter até 15 endpoints de um dispositivo (0 é o de controle). Depois do token ser recebido pelo dispositivo, a transmissão de dados é gerada pelo host ou pelo dispositivo dependendo da direção especificada. Uma vez que a transmissão de dados é completada, é gerado um pacote handshake. O pacote ACK é gerado pelo dispositivo se o host transmitiu, ou pelo host se foi enviado dados para ele.

Protocolo USB

- O USB suporta trocas de dados e controles entre o host e dispositivos USB através de um conjunto de canais (pipes) lógicos. Um pipe é a associação entre um endpoint do periférico e um software do host, do ponto de vista do software há uma conexão direta onde se é mascarado/escondido os detalhes de hierarquia do barramento.
- Há dois tipos de pipes: o stream pipe que não possui uma estrutura definida, e o message pipe cuja estrutura é definida pelo protocolo.

Protocolo USB

- Sempre que um dispositivo é conectado um Default Control Pipe é estabelecido com o intuito de acessar a configuração, status e informações de controle.
- Com a finalidade de suportar muitos tipos diferentes de dispositivos e usos, são definidos quatro tipos fundamentais de transferências de dados:

Protocolo USB

- Control transfers: usados para configurar os dispositivos quando são ligados ao barramento.
- Bulk data transfers: usados em transferência de grandes quantidade de dados de forma sequencial.
- Interrupt Data Transfers: usado para transmitir/receber pequenas quantidades de dados em baixa velocidade, de tempo em tempo.
- Isochronous Data Transfer: usados para transferência de grandes quantidades de dados, mas com largura de banda e latência pré-negociadas, e tolerante a erros.

Codificação/decodificação NRZI

- A codificação/decodificação NRZI consiste em definir ‘0’ ou ‘1’ pela mudança de nível de tensão nas linhas. Em que ‘1’ é representado quando não há mudança de nível e ‘0’ quando há mudança de nível.

Módulo SSP

- Módulo compatível com SPI (Motorola), 4 wire da Texas e Microware da National, estes dois último podem ser considerados como uma das formas de trabalho da SPI.
- 4 wire – Texas Instrument – pinos: clock, entrada de dados (DR) e saída de dados (DX) e o pino Frame Sync (FR) que identifica o término da informação.
- Microware – National Semiconductor – pinos: clock, dados in (SI) e dados out (SO) e chip select (CS), porém só há envio ou recebimento de dado de cada vez.

LIN - Local Interconnect Network

- Desenvolvido para a indústria automobilística. Trata-se de um barramento simples, de baixa velocidade (20 kbps), utiliza apenas um fio para conexão entre os elementos de comunicação em uma arquitetura mestre-escravo.

LIN - Local Interconnect Network

- O CAN é um protocolo com maior velocidade que o LIN, o LIN é um protocolo para ser utilizado usando as USART das MCUs.
- Algumas características são:
- Falta de arbitragem (só há um mestre)
- Garantia de tipo de latência
- Frames flexíveis 2,4, ou 8 bits.
- Detecção de erros por checksum.

Voltando ao Tiva

- Módulo UART!

UART – Universal Asynchronous Receiver Transmitter

- Modo assíncrono: comunicação feita em duas vias, uma para transmissão (TX) e outra para recepção (RX), logo é full-duplex. Usada para implementar o padrão TIA/EIA 232. A sincronia é feita através da configuração do baud rate nos dispositivos e por bits de start e stop.

Características do Tiva

- Gerador de baudrate programável de até 5Mbps ou então 10 Mbps;
- FIFOs separadas de 16x8 para TX e RX;
- Comprimento programável da FIFO;
- Níveis de trigger de FIFO: 1/8, 1/4, ½, ¾ e 7/8;
- Bits padrão para start, stop e paridade;
- Características programáveis:
  - 5,6,7 ou 8 bits de dados;
  - Paridade: par, ímpar ou sem paridade;
  - 1 ou 2 stop bits.

Pinos com sinal de UART

Pin Name	Pin Number	Pin Mux / Pin Assignment	Pin Type	Buffer Type[a]	Description
U0Rx	17	PA0 (1)	I	TTL	UART module 0 receive.
U0Tx	18	PA1 (1)	O	TTL	UART module 0 transmit.
U1CTS	15 29	PC5 (8) PF1 (1)	I	TTL	UART module 1 Clear To Send modem flow control input signal.
U1RTS	16 28	PC4 (8) PF0 (1)	O	TTL	UART module 1 Request to Send modem flow control output line.
U1Rx	16 45	PC4 (2) PB0 (1)	I	TTL	UART module 1 receive.
U1Tx	15 46	PC5 (2) PB1 (1)	O	TTL	UART module 1 transmit.
U2Rx	53	PD6 (1)	I	TTL	UART module 2 receive.
U2Tx	10	PD7 (1)	O	TTL	UART module 2 transmit.
U3Rx	14	PC6 (1)	I	TTL	UART module 3 receive.
U3Tx	13	PC7 (1)	O	TTL	UART module 3 transmit.
U4Rx	16	PC4 (1)	I	TTL	UART module 4 receive.
U4Tx	15	PC5 (1)	O	TTL	UART module 4 transmit.
U5Rx	59	PE4 (1)	I	TTL	UART module 5 receive.
U5Tx	60	PE5 (1)	O	TTL	UART module 5 transmit.
U6Rx	43	PD4 (1)	I	TTL	UART module 6 receive.
U6Tx	44	PD5 (1)	O	TTL	UART module 6 transmit.
U7Rx	9	PE0 (1)	I	TTL	UART module 7 receive.
U7Tx	8	PE1 (1)	O	TTL	UART module 7 transmit.

Modo 9 Bits de UART

- Modo utilizado para conectar um único master a múltiplos escravos.

Interrupções

- A UART pode gerar interrupções quando as seguintes condições são observadas:
- Overrun error
- Break error
- Parity error
- Framing Error
- Receive Timeout
- Transmit
- Receive

Inicialização e configuração

- Habilitar o módulo UART;
- Habilitar o módulo GPIO dos pinos envolvidos;
- Selecionar a funcionalidade dos pinos como UART;
- Configurar os níveis de corrente dos pinos;
- Designar os pinos para o módulo UART;
- Configurar o baudrate, tamanho dos dados, bits de stop e paridade.

Inicialização e configuração: Interrupção

- Além das configurações anteriores:
- Habilitar o processador de interrupções;
- Incluir no startup ou utilizar a função UARTIntRegister para associar a função tratadora da interrupção;
- Habilitar a interrupção do módulo;
- Configurar as ações que irão disparar a interrupção.

Anderson Wedderhoff Spengler

E-mail: anderson.spengler@ufsc.br

Telefone: +55 (48) 3721 7489

UNIVERSIDADE FEDERAL
DE SANTA CATARINA