5197 - Sistema Digitais

Bacharelado de Informática

UEM – DIN - Prof. Elvio

2016

Roteiro

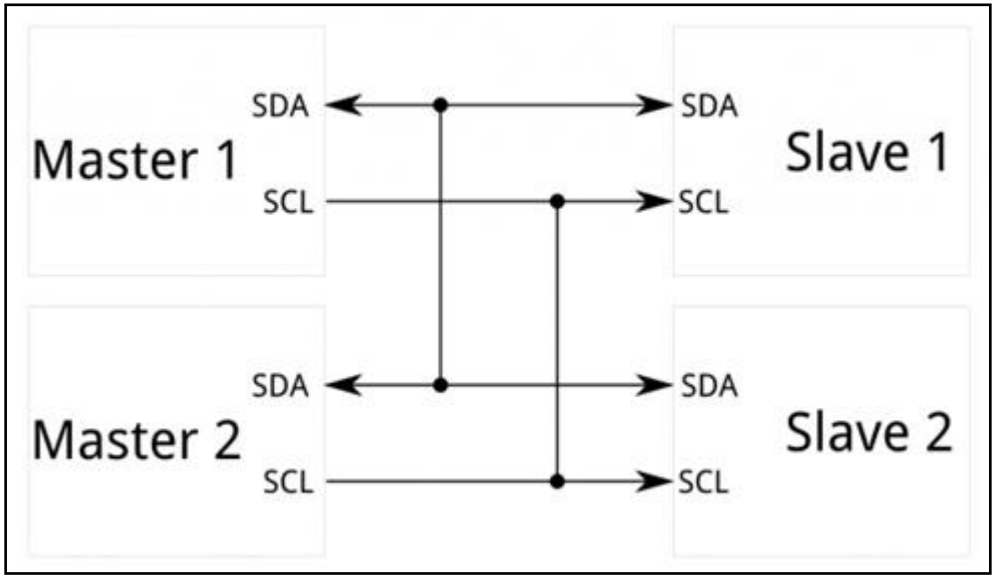
- TWI (*Twin Wire Interface*)

Protocolo I^2C

- Protocolo I^2C (*Inter-Integrated Circuit*) foi desenvolvido pela Philips na década de 80 para comunicação entre circuitos integrados
- Permite a comunicação de **múltiplos mestres** e **múltiplos escravos** em um único barramento
- Semelhanças com outros sistemas
  - Assim como o SPI (*Serial Peripheral Interface*), é utilizado para comunicação síncrona a pequenas distâncias
  - Assim como o RS232, a comunicação é feita com apenas 2 fios
- Diferenças de outros sistemas
  - Ao contrário do SPI, apenas 2 pinos são necessários
  - Ao contrário do RS232, a comunicação é síncrona

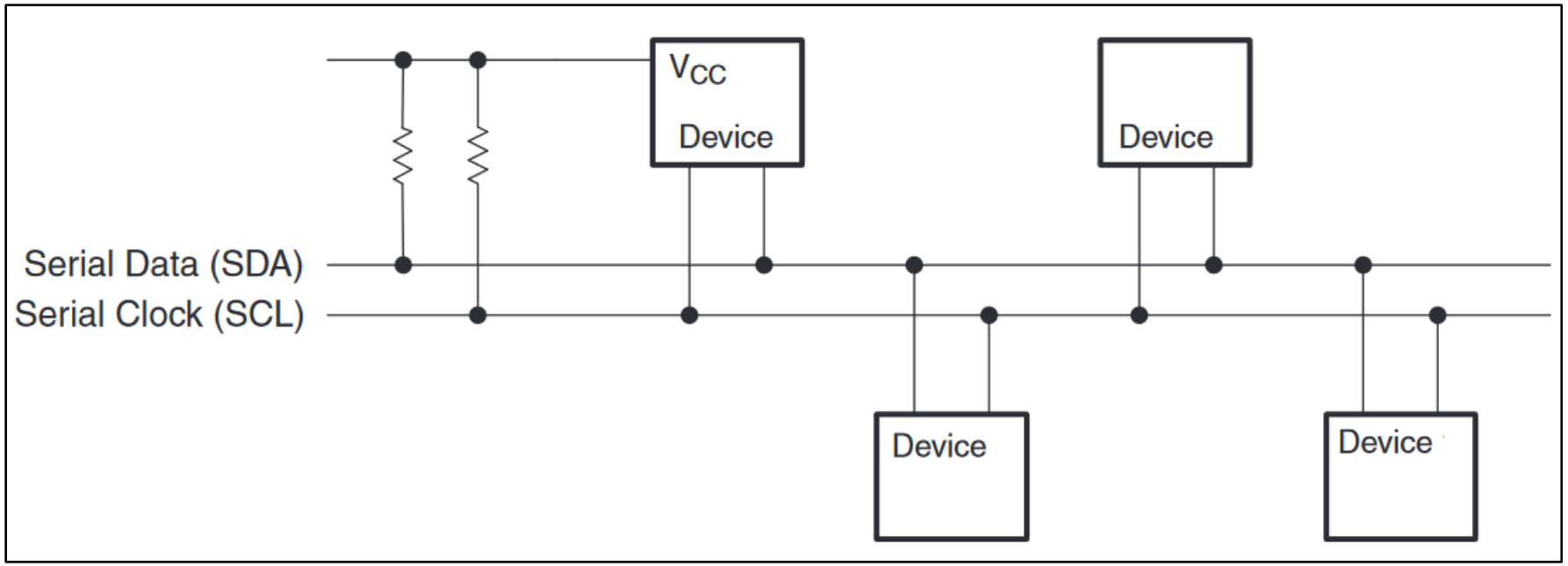
Protocolo I^2C

- Portanto o I^2C oferece as vantagens dos dois sistemas: comunicação síncrona com apenas 2 fios
- Endereçamento com 7 *bits* ou 10 *bits* **por software**
- Relógios de 100 kHz (*standard mode*), 400 kHz (*fast mode*), 1 MHz (*fast-mode plus*) 3,4 MHz (*high-speed mode*) e 5 MHz (*ultra-fast mode*)



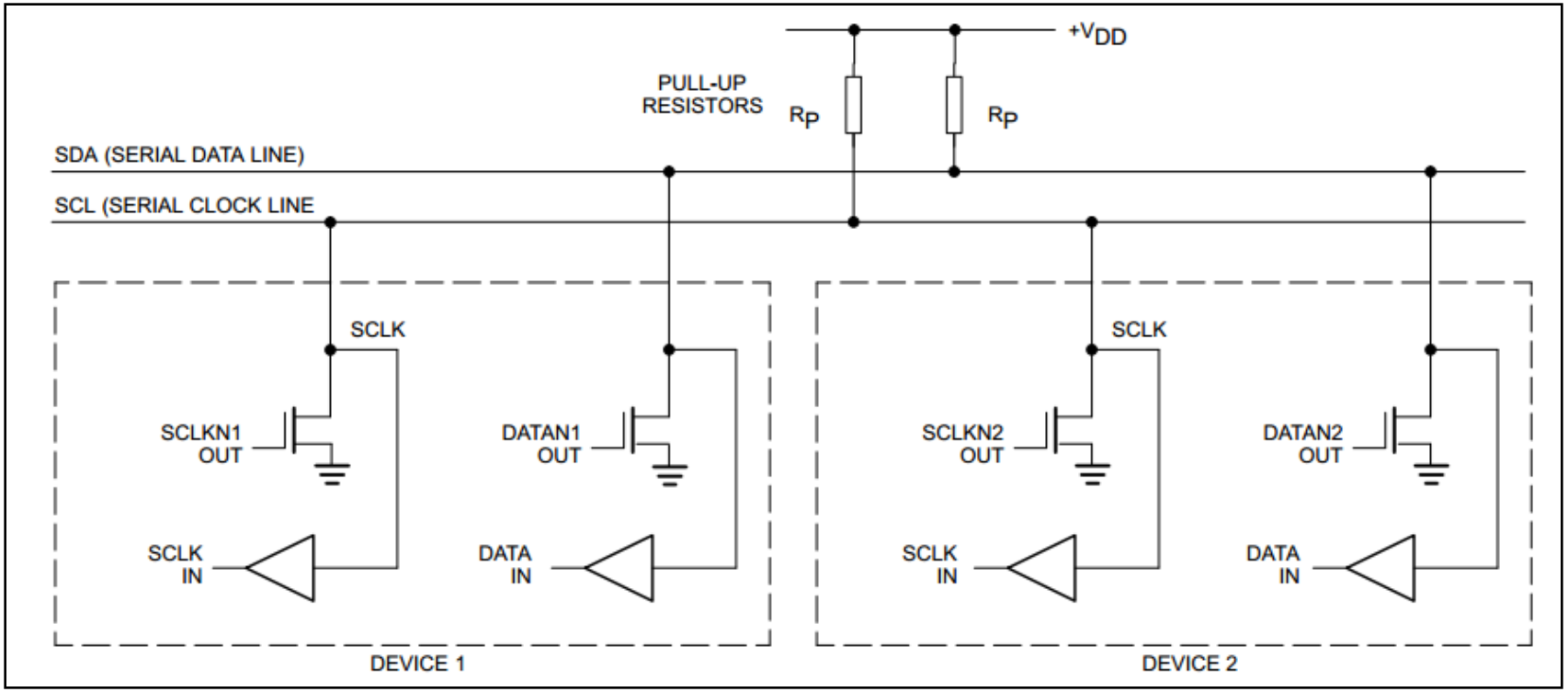
Protocolo I^2C

- Mestre é responsável pela geração do relógio do barramento
- Barramento composto de sinais SDA e SCL



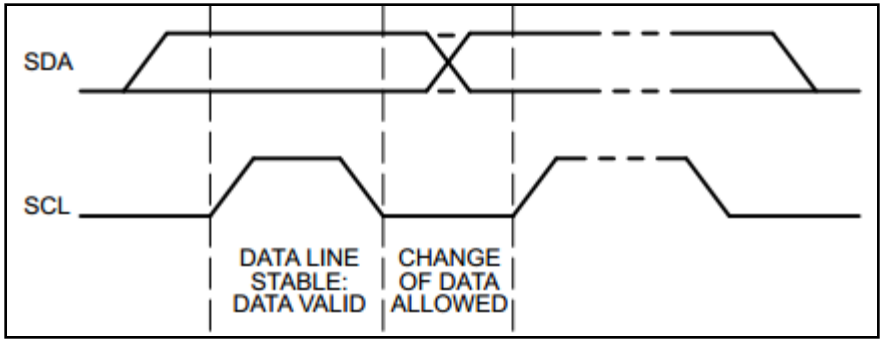
Protocolo I^2C

- Nova transmissão somente acontece se barramento está livre
- Interface implementa *wired*-AND

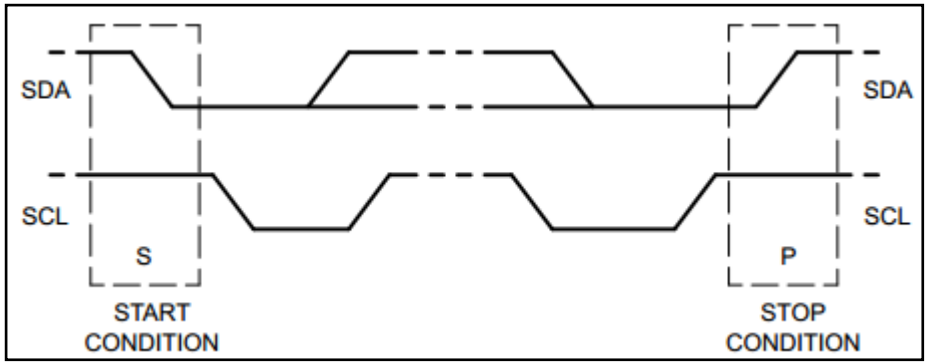


Protocolo I^2C

- Amostragem dos dados
  - Realizada durante o nível “1” do sinal de relógio

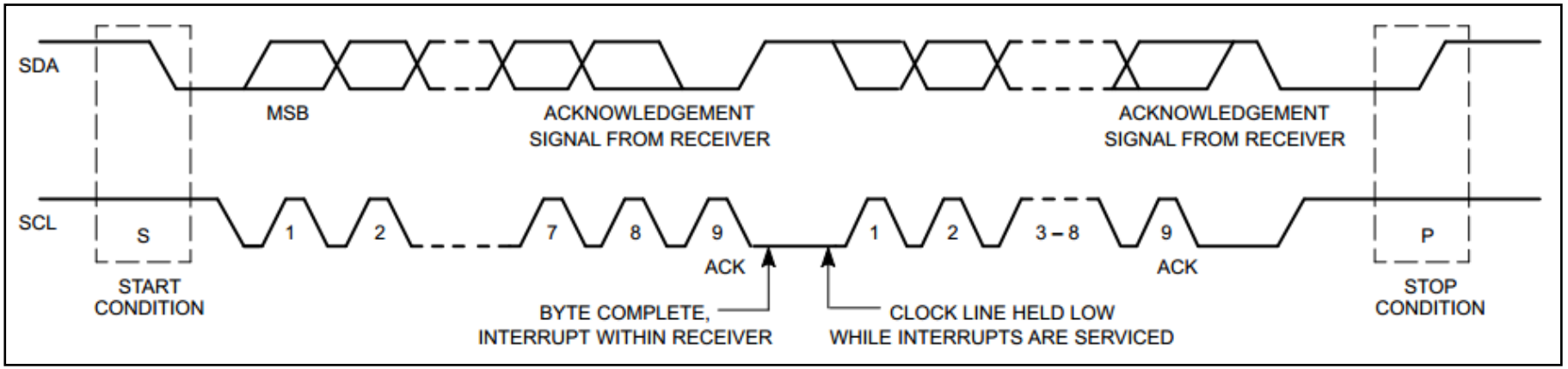


- Condições START e STOP

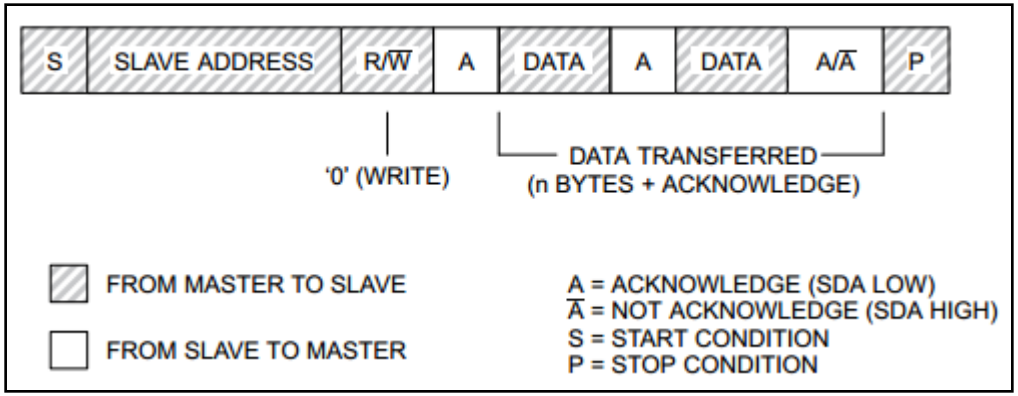


Protocolo I²C

- Transferência de dados genérica

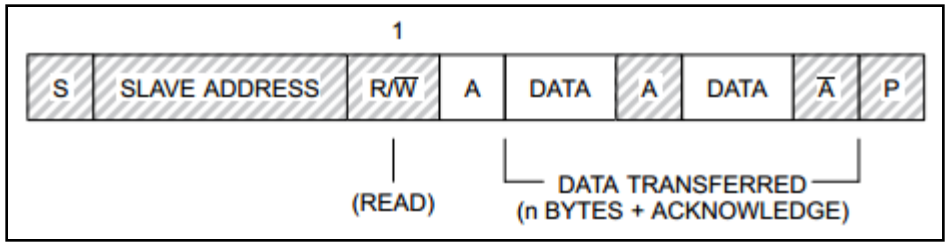


- Exemplo 1: Escrita pelo mestre

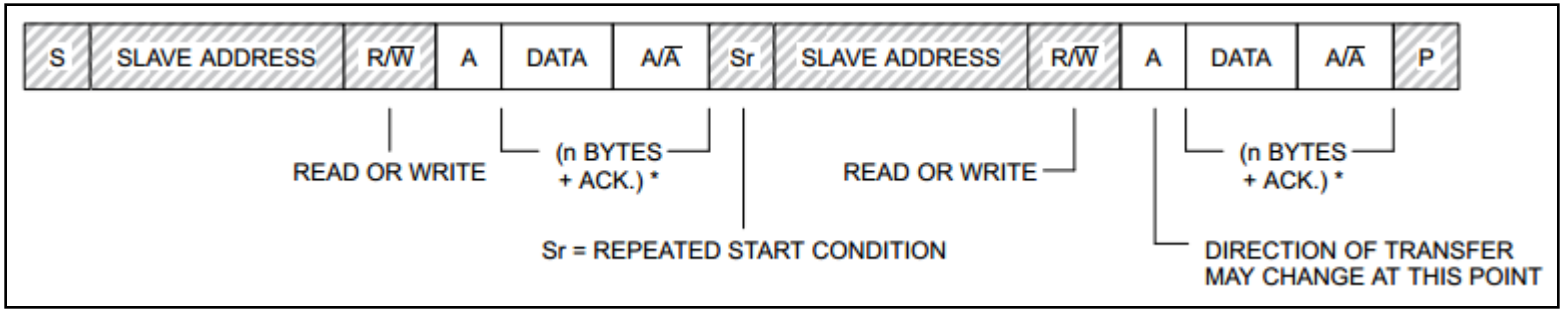


Protocolo I²C

- Exemplo 2: Leitura pelo mestre

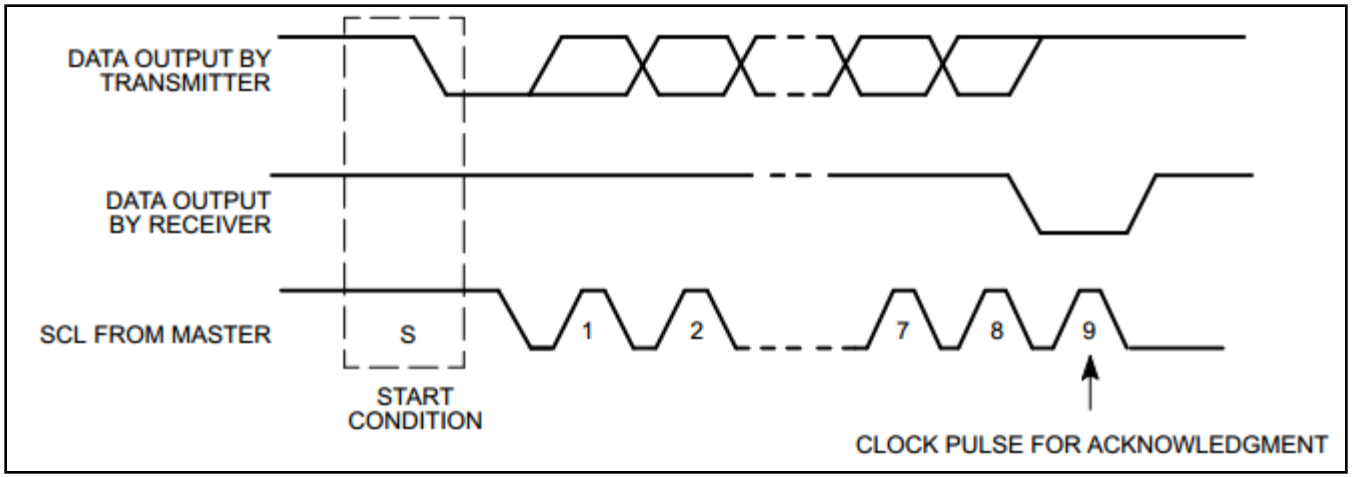


- Exemplo 3: Leitura e escrita combinadas



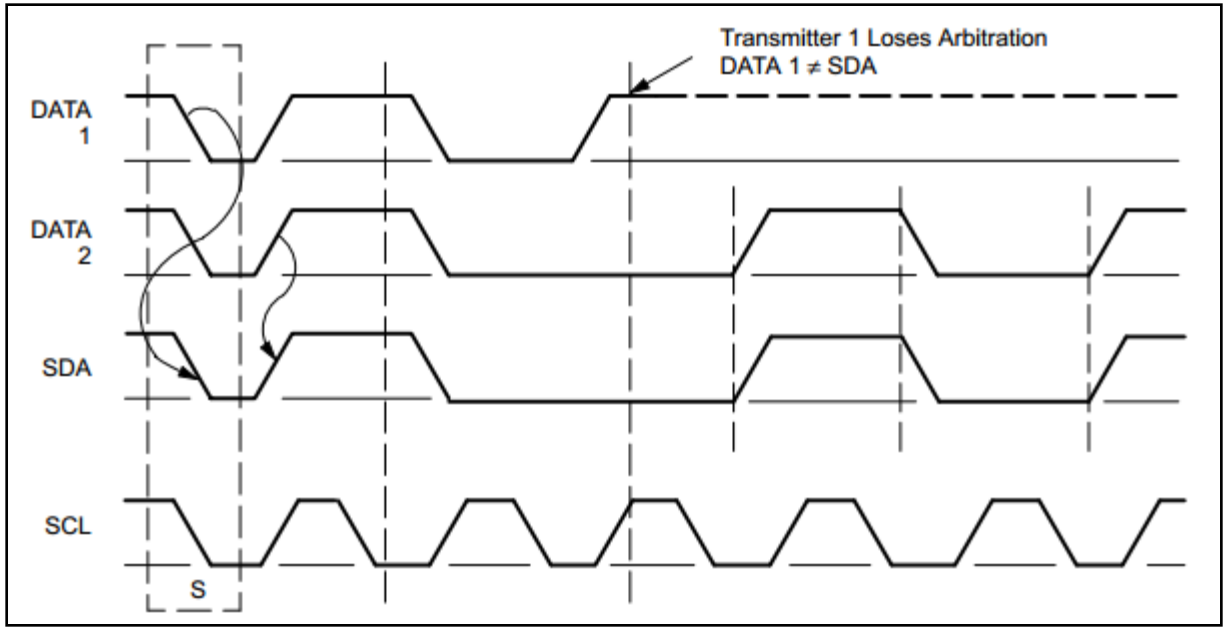
Protocolo I^2C

- Bit de *Acknowledgement*
  - Confirma o recebimento de dados
  - Preenchido pelo receptor (mestre ou escravo)



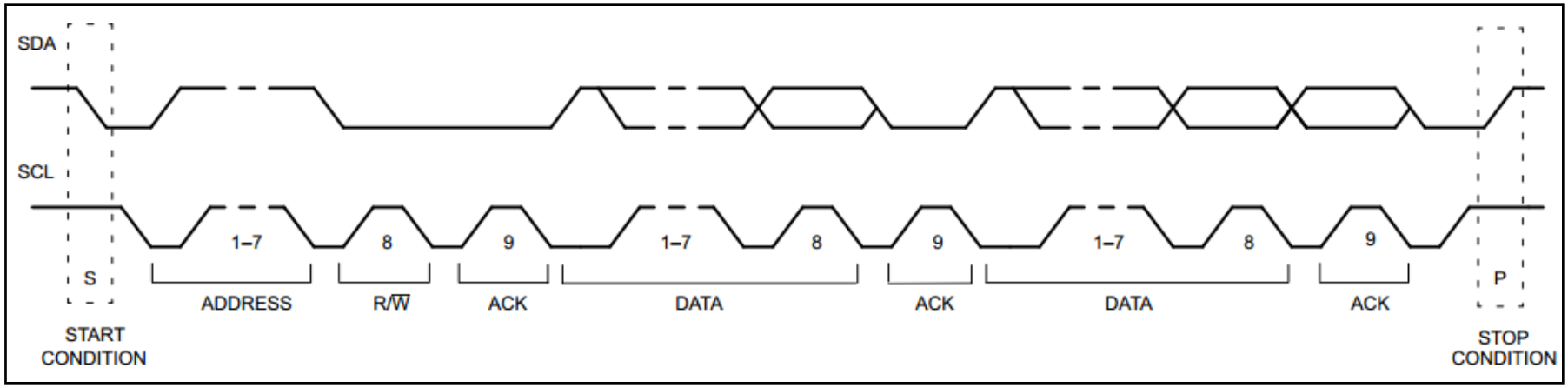
Protocolo I^2C

- Arbitragem (*Arbitration*)
  - Quando dois mestres querem transmitir ao mesmo tempo



Protocolo I^2C

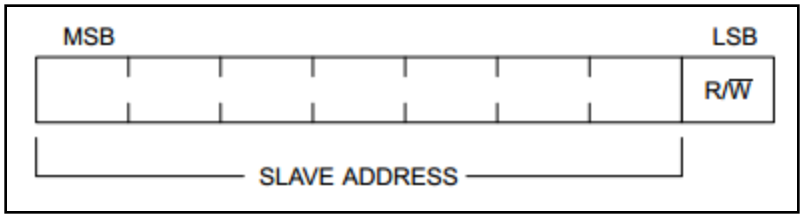
- Exemplo de ciclo de transferência completo



  - Controle de fluxo:
    - Se, após o 1º *byte*, o receptor não puder receber outros, ele deve manter SCL baixo para forçar o transmissor entrar em estado de espera
    - Transferência continua após o receptor liberar o SCL

Protocolo I²C

- Endereçamento de 7 *bits*
  - Primeiro *byte* da transferência



  - Endereços reservados

SLAVE ADDRESS	R/ bit	DESCRIPTION
0000 000	0	General call address
0000 000	1	START byte
0000 001	X	CBUS address
0000 010	X	Address reserved for different bus format
0000 011	X	Reserved for future purposes
0000 1XX	X	
1111 1XX	X	
1111 0XX	X	10-bit slave addressing

Protocolo I^2C

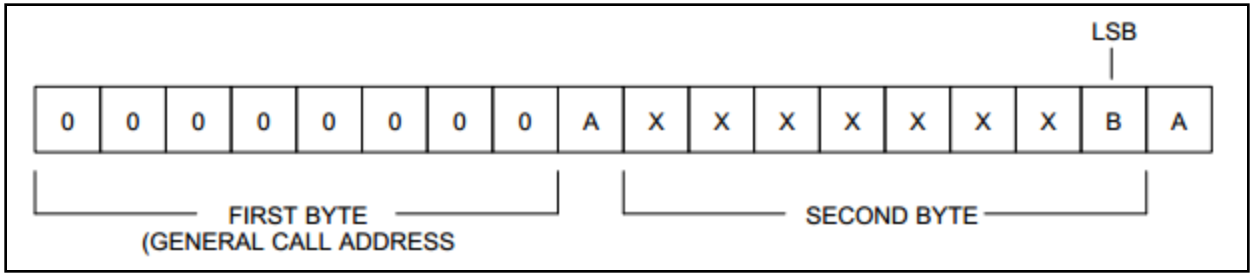
- Endereços reservados
  - *General Call Address*: Endereço de *broadcast* da rede
    - Dispositivos que não precisam de dados de *broadcast* podem ignorar e deixar de marcar o *bit* de *acknowledgement*
    - Dispositivos que aceitem os dados devem marcar o *bit* de *acknowledgement* e comportar-se como receptor escravo
    - Significado do *General Call* é determinado pelo 2º *byte*:
      - Se o bit menos significativo do segundo *byte* é 0 (B = 0)
      - Se o bit menos significativo do segundo *byte* é 0 (B = 1)

Protocolo I^2C

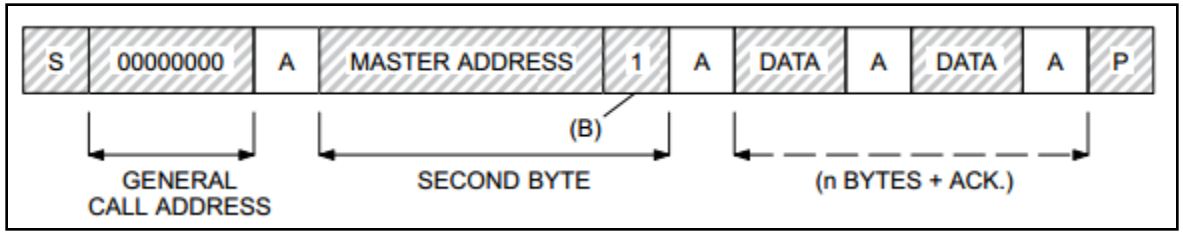
- Endereço *General Call Address*
  - Para B = 0
    - 2º *byte* = 00000110B: *reset* e carrega a parte programável do endereço de escravo
    - 2º *byte* = 00000100B: carrega parte programável do endereço de escravo (sem *reset*)
    - 2º *byte* = 00000000B: não é permitido
  - Para B = 1:
    - Representa uma *Hardware General Call*
    - Usado por dispositivos que não sabem ou não conseguem determinar qual o endereço do escravo para o qual querem transmitir (por exemplo, um teclado)
    - Dispositivo inclui o seu próprio endereço no 2º *byte* do *General Call*
    - Este endereço é reconhecido pelo dispositivo inteligente (por exemplo, um microprocessador) que então recebe os dados

Protocolo I²C

- Formato do endereço do *General Call*

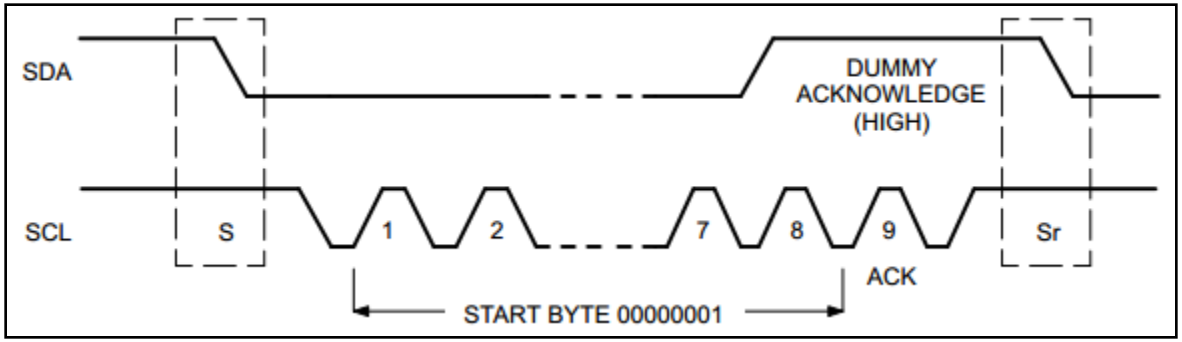


- Ciclo de transferência do HW *General Call*



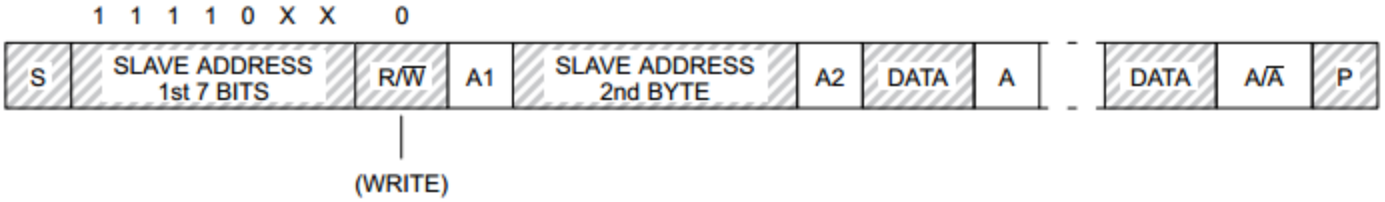
Protocolo I^2C

- Endereço *Start Byte*
  - Utilizado para atrasar o início da transferência de dados
    - Útil para microcontroladores que não têm hardware dedicado à interface e precisam fazer *polling* dos sinais
  - Endereço *Start Byte* não deve ser ter o *bit* de *acknowledgement* marcado

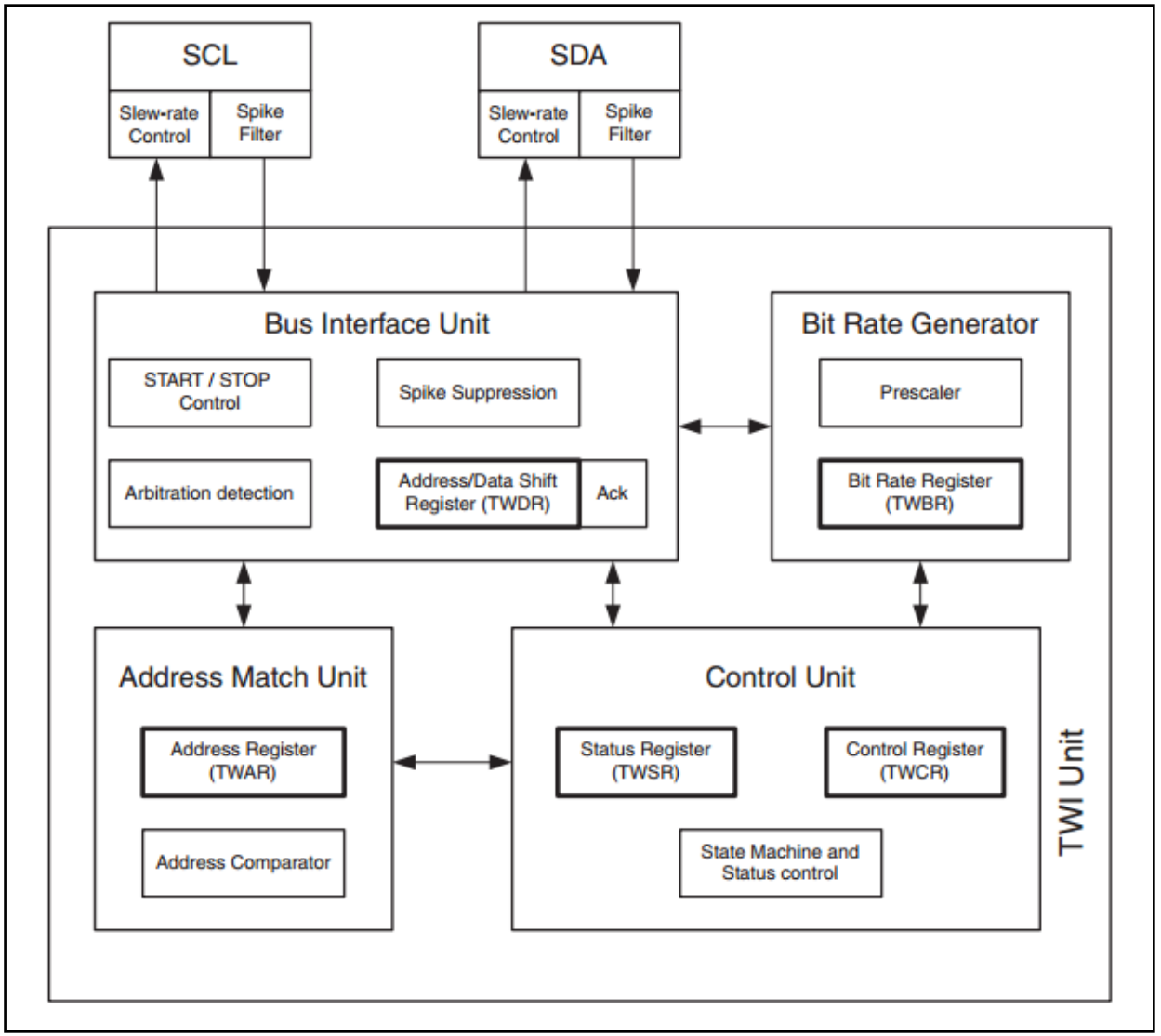


Protocolo I^2C

- Endereço CBUS
  - Utilizado para interconexão com dispositivos CBUS
- Endereço de 10 *bits*



I^2C (ou TWI) no ATmega328



I²C (ou TWI) no ATmega328

- Determinação da frequência do SCL

$$\text{SCL frequency} = \frac{\text{CPU Clock frequency}}{16 + 2(\text{TWBR}) \cdot (PrescalerValue)}$$

- Registrador TWI *Bit Rate* (TWBR)

Bit	7	6	5	4	3	2	1	0
(0xB8)	TWBR7	TWBR6	TWBR5	TWBR4	TWBR3	TWBR2	TWBR1	TWBR0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

  - TWBR0-7: seleciona o fator de divisão do gerador de taxa de *bit*

I²C (ou TWI) no ATmega328

- Registrador TWI *Control* (TWCR)

Bit	7	6	5	4	3	2	1	0
(0xBC)	TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R	R/W
Initial Value	0	0	0	0	0	0	0	0

  - TWI *Interrupt Flag* (TWINT): indicador de interrupção, setado pelo HW ao final da tarefa corrente
  - TWI *Enable Acknowledge Bit* (TWEA): habilita marcar o *bit de acknowledgment*
  - TWI START *Condition Bit* (TWSTA): interface gera uma Condição START

I^2C (ou TWI) no ATmega328

- Registrador TWI *Control* (TWCR)
  - TWI STOP *Condition Bit* (TWSTO): interface gera uma Condição STOP
  - TWI *Write Collision Flag* (TWWC): indica escrita no registrador de dados com o *bit* de interrupção em 0 (deve-se limpar o bit de interrupção por último)
  - TWI *Enable Bit* (TWEN): habilita a interface TWI
  - TWI *Interrupt Enable* (TWIE): habilita a interrupção

I²C (ou TWI) no ATmega328

- Registrador TWI *Status* (TWSR)

Bit	7	6	5	4	3	2	1	0
(0xB9)	TWS7	TWS6	TWS5	TWS4	TWS3	–	TWPS1	TWPS0
Read/Write	R	R	R	R	R	R	R/W	R/W
Initial Value	1	1	1	1	1	0	0	0

  - TWI Status (TWS): indica o status da interface
  - TWI *Prescaler Bits* (TWPS): multiplicador da frequência da SCL

TWPS1	TWPS0	Prescaler Value
0	0	1
0	1	4
1	0	16
1	1	64

I²C (ou TWI) no ATmega328

- Registrador TWI *Data* (TWDR)

Bit	7	6	5	4	3	2	1	0
(0xBB)	TWD7	TWD6	TWD5	TWD4	TWD3	TWD2	TWD1	TWD0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	1	1	1	1	1	1	1	1

- Registrador TWI *Slave Address* (TWAR)

Bit	7	6	5	4	3	2	1	0
(0xBA)	TWA6	TWA5	TWA4	TWA3	TWA2	TWA1	TWA0	TWGCE
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	1	1	1	1	1	1	1	0

  - TWI *General Call Recognition Enable Bit* (TWGCE): habilita a detecção da *General Call*

I²C (ou TWI) no ATmega328

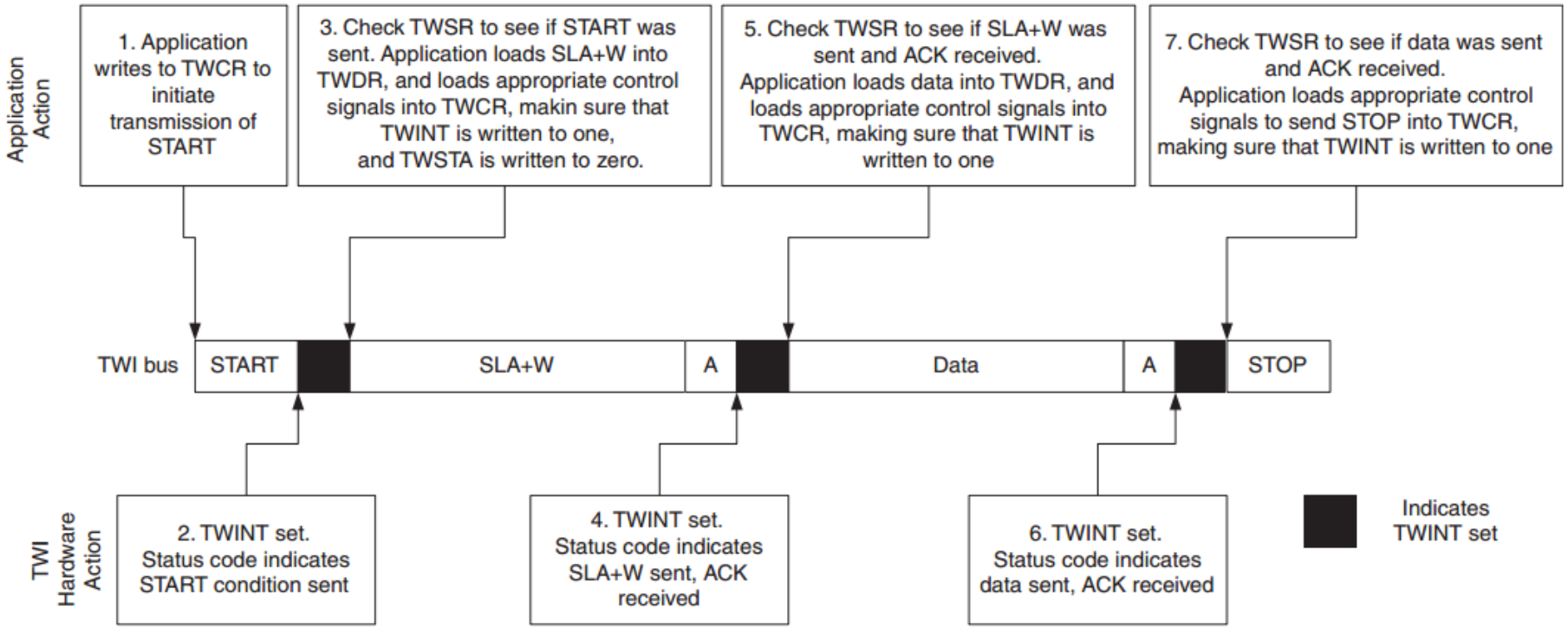
- Registrador TWI *Slave Address Mask* (TWAMR)

Bit	7	6	5	4	3	2	1	0
(0xBD)	TWAM[6:0]							–
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Initial Value	0	0	0	0	0	0	0	0

  - Mascara (desabilita) o bit correspondente do endereço do escravo armazenado em TWAR

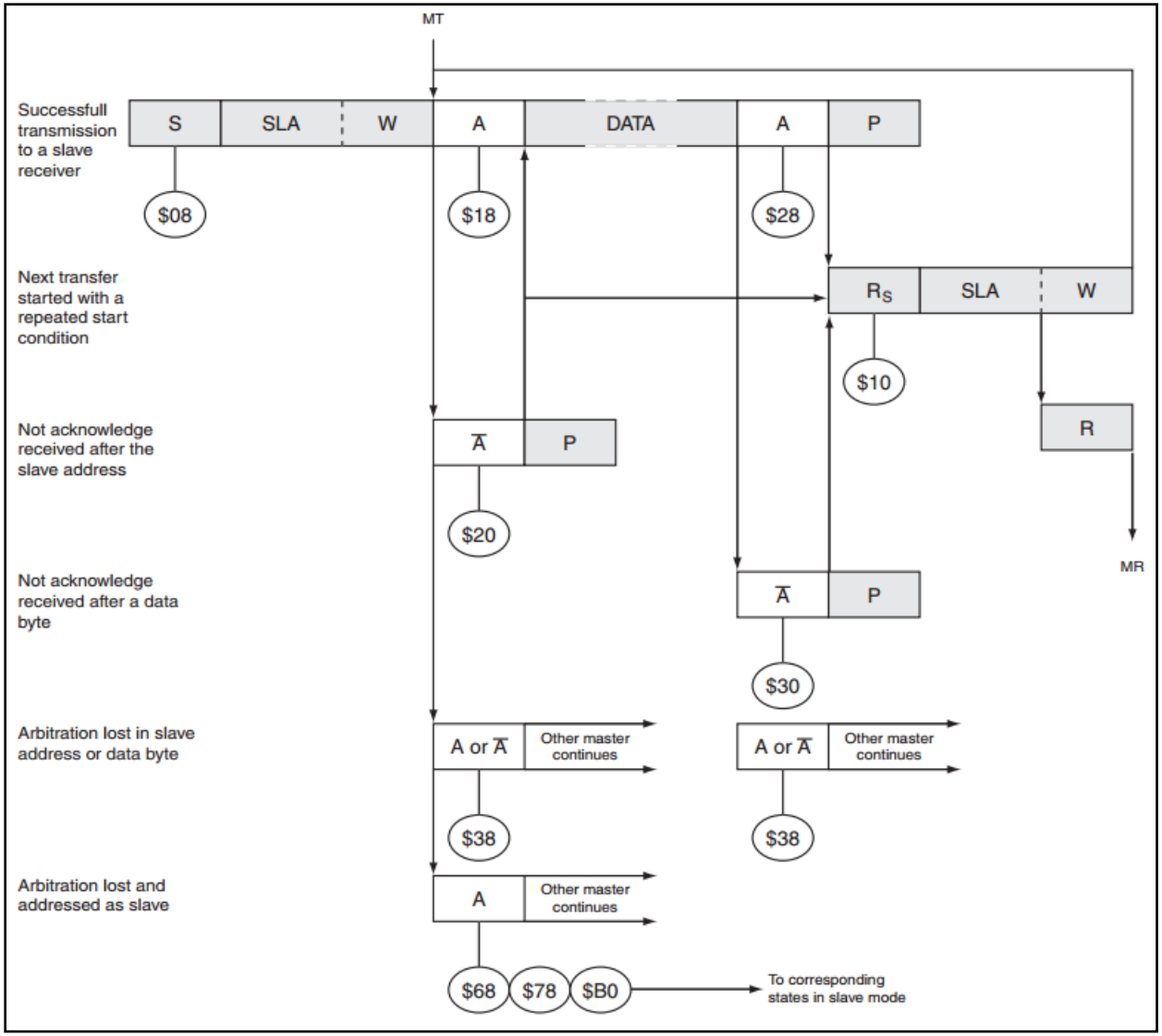
I^2C (ou TWI) no ATmega328

- Transmissão usando a TWI



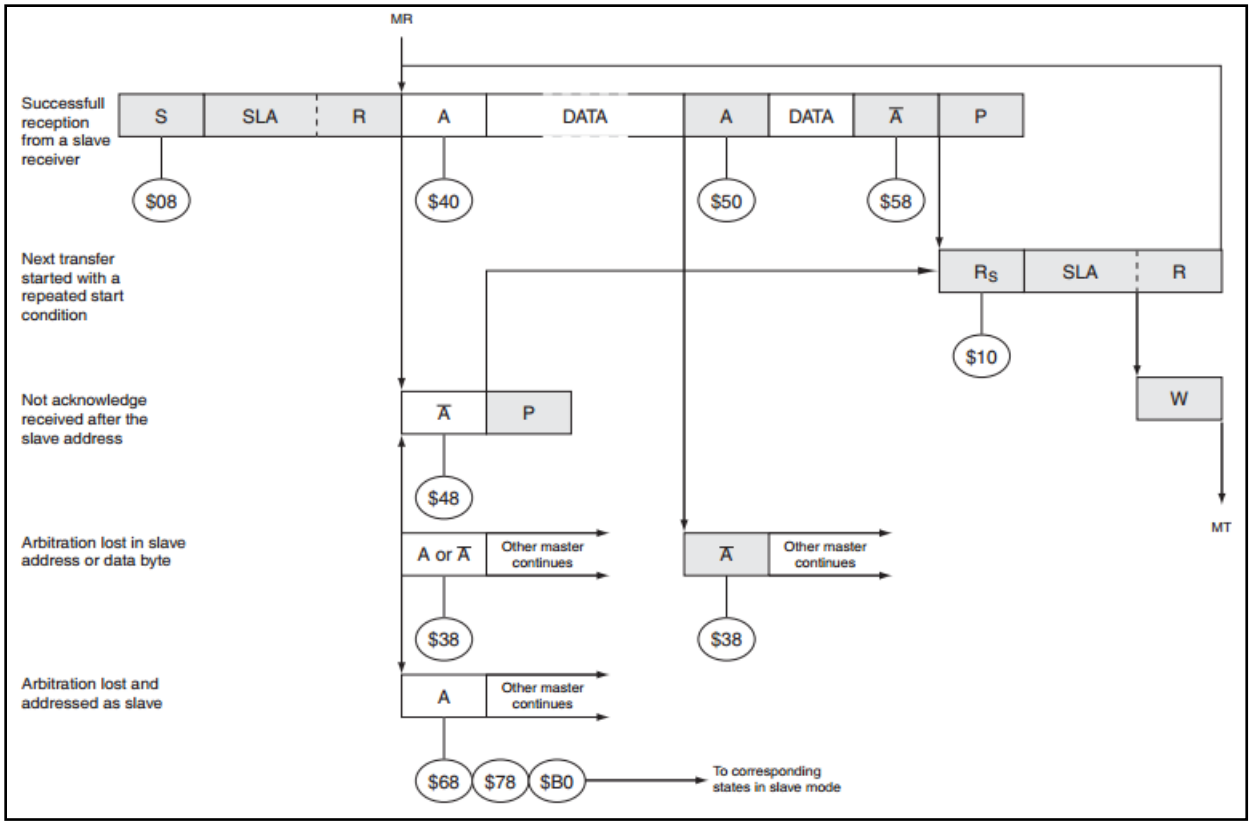
I^2C (ou TWI) no ATmega328

- Estados para transmissão como mestre



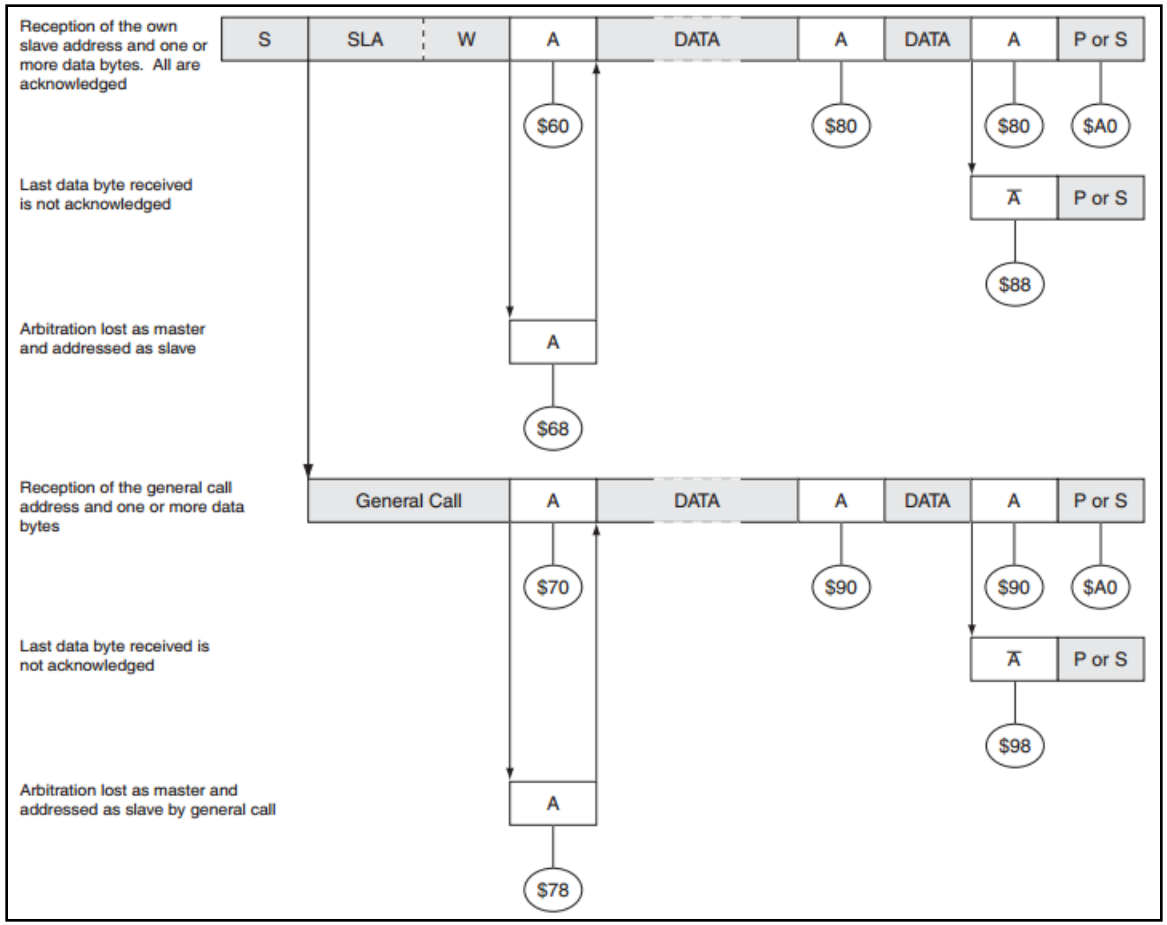
I^2C (ou TWI) no ATmega328

- Estados para recepção como mestre



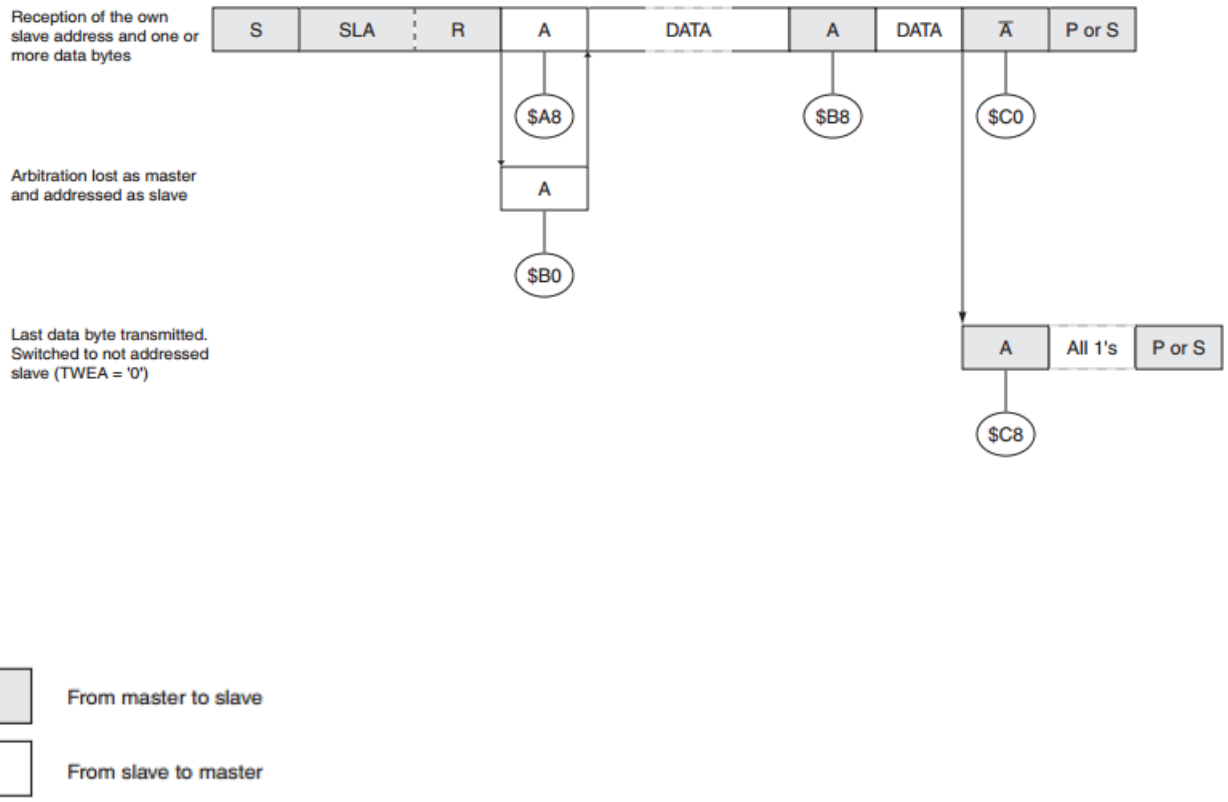
I^2C (ou TWI) no ATmega328

- Estados para recepção como escravo



I^2C (ou TWI) no ATmega328

- Estados para transmissão como escravo



I^2C (ou TWI) no Arduino

```
//=============================================================================//
//                    CÓDIGO EXEMPLO PARA USO DO TWI                           //
//=============================================================================//
    TWCR = (1<<TWINT)|(1<<TWSTA)|(1<<TWEN);//Envia a condição de início
//-----------------------------------------------------------------------------
    while (!(TWCR & (1<<TWINT)));   //Espera o TWINT ser ativo indicando que a
                                    //condição de início foi transmitida
//-----------------------------------------------------------------------------
    if ((TWSR & 0xF8) != START)     //Verifica o valor do TWI no registrador de status.
        ERROR();                    /*Mascara os bits do prescaler. Se o status
                                    for diferente da condição de início chama
                                    uma função para o tratamento do erro*/
//-----------------------------------------------------------------------------
    TWDR = SLA_W;                   //Carrega o endereço do escravo para a escrita
    TWCR = (1<<TWINT) | (1<<TWEN);  /*limpa o bit TWINT no TWCR para começar a
                                    transmissão do endereço*/
//-----------------------------------------------------------------------------
    while (!(TWCR & (1<<TWINT)));   /*Espera pela ativação do bit TWINT
                                    indicando que o endereço do escravo + o bit de
                                    escrita foi enviado e que o ACK/NACK foi recebido*/
```

I^2C (ou TWI) no Arduino

```
//-------------------------------------------------------------------------------
    if ((TWSR & 0xF8) != MT_SLA_ACK)//Verifica o valor do registrador de
        ERROR();                    /*status do TWI. Mascara os bits do prescaler. Se
                                    o status for diferente de MT_SLA_ACK chama uma
                                    função para o tratamento do erro*/
//-------------------------------------------------------------------------------
    TWDR = DATA;    //Carrega DATA no registrador TWDR. Limpa o bit TWINT no TWCR
    TWCR = (1<<TWINT) | (1<<TWEN); //para iniciar a transmissão do dado.
//-------------------------------------------------------------------------------
    while (!(TWCR & (1<<TWINT)));/*Espera o bit TWINT ser ativo, indicando que
                                 o dado foi transmitido e que o ACK/NACK foi recebido*/
//-------------------------------------------------------------------------------
    if ((TWSR & 0xF8) != MT_DATA_ACK)    //Verifica o valor do registrador de
         ERROR();                  /*status do TWI. Mascara os bits do prescaler. Se o
                                   status for diferente de MT_DATA_ACK chama uma função
                                   para o tratamento do erro*/
//-------------------------------------------------------------------------------
    TWCR = (1<<TWINT)|(1<<TWEN)| (1<<TWSTO);//Transmite a condição de parada
//===============================================================================
```