Politechnika Wrocławska

Systemy RT i embedded

Wykład 9
Interfejsy mikrokontrolerów, cz. I

Wrocław 2013

Plan

- Microncontrollers' interfaces
- SCI
- SPI
- I^2C
- OneWire
- I^2S

Microcontrollers' interfaces

Types of interfaces

Interface:

a. equipment or programs designed to communicate information from one system of computing devices or programs to another.

b. any arrangement for such communication.

Types of interfaces

- Types:
  - Software
  - Hardware

  - Serial
  - Parallel:
    - 4-bit
    - 8-bit
    - 16-bit
    - …

Serial vs parallel

- Features of serial interfaces:
  - Simplicity:
    - Medium (RS232)
    - Large (OneWire)
  - Perfomance
    - From very small (OneWire) to very large (Ethernet)
  - Implementation difficulties:
    - From small (RS232) to large (Bluetooth)

Serial vs parallel

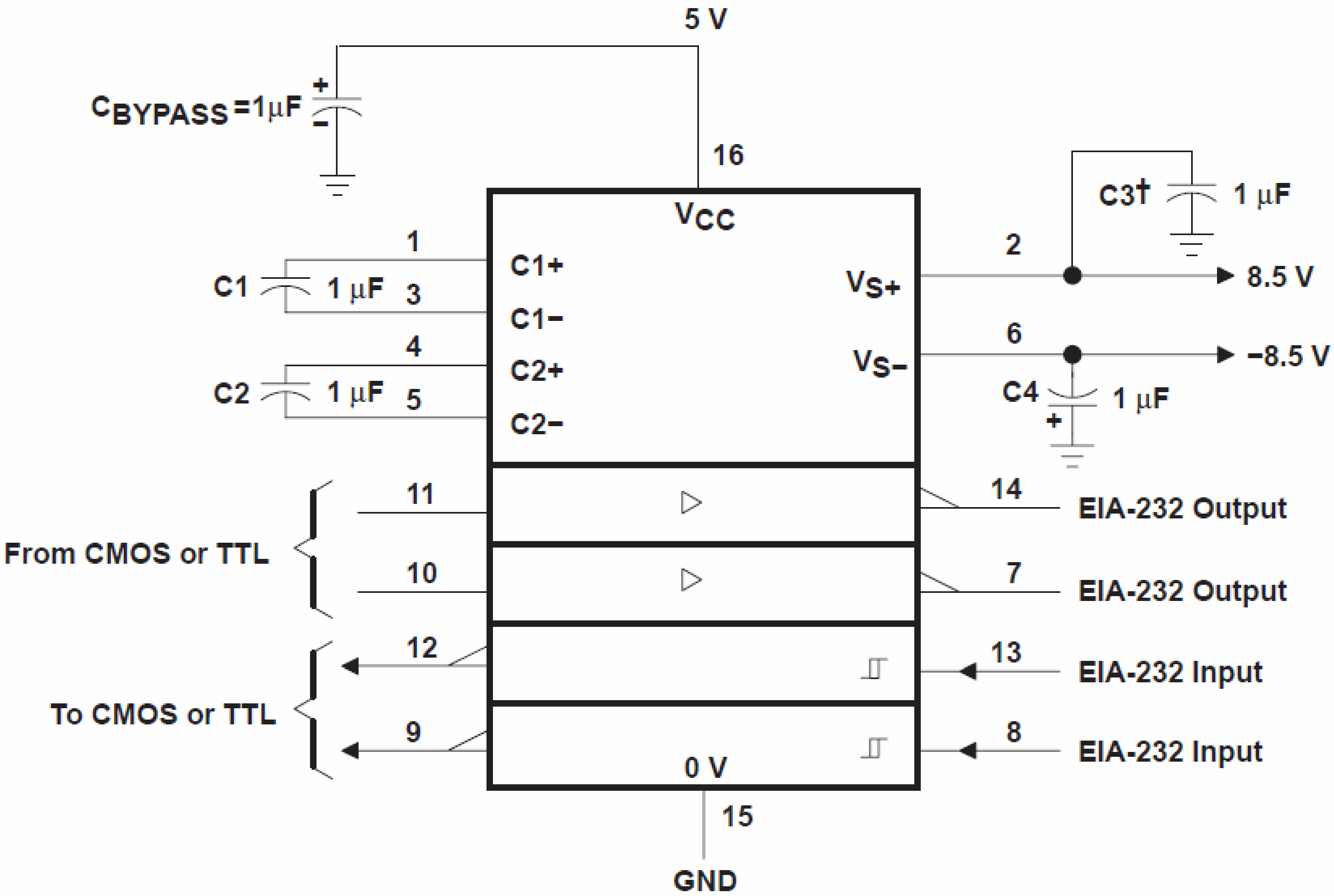
- Features of parallel interfaces:
  - Simplicity:
    - Medium
  - Perfomance
    - Large or very large (64-bit)
    - Limited by propagation delay, noises
  - Implementation difficulties:
    - Small at low speed
    - Very large at high speeds
    - Problems with large range implmentations

UART interface

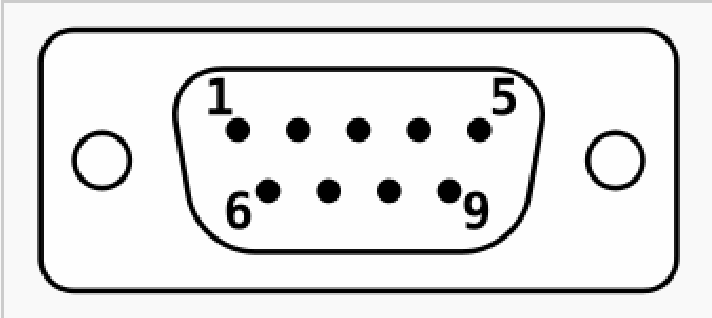
UART interface

- Features:
  - One of the most popular serial interfaces
  - Serial, two-wire interface
  - Used for internal and external communication
  - Original versions (e.g. RS-232C) require many control lines (DB9, DB25)
  - Two state logic used, but not TTL!
  - Asynchronous data transfer. **No clock** transmitted!
  - Full duplex transmission
  - Point – to – point transmission
  - Throughput up to hundreds of kb/s

MAX232


5 V
CBYPASS =1µF
+
−
16
VCC
1
C1
1 µF
3
C1+
C1−
4
C2
1 µF
5
C2+
C2−
2
C3†
1 µF
VS+
8.5 V
6
VS−
−8.5 V
C4
1 µF
+
11
14
EIA-232 Output
From CMOS or TTL
10
7
EIA-232 Output
12
13
EIA-232 Input
To CMOS or TTL
9
8
EIA-232 Input
0 V
15
GND

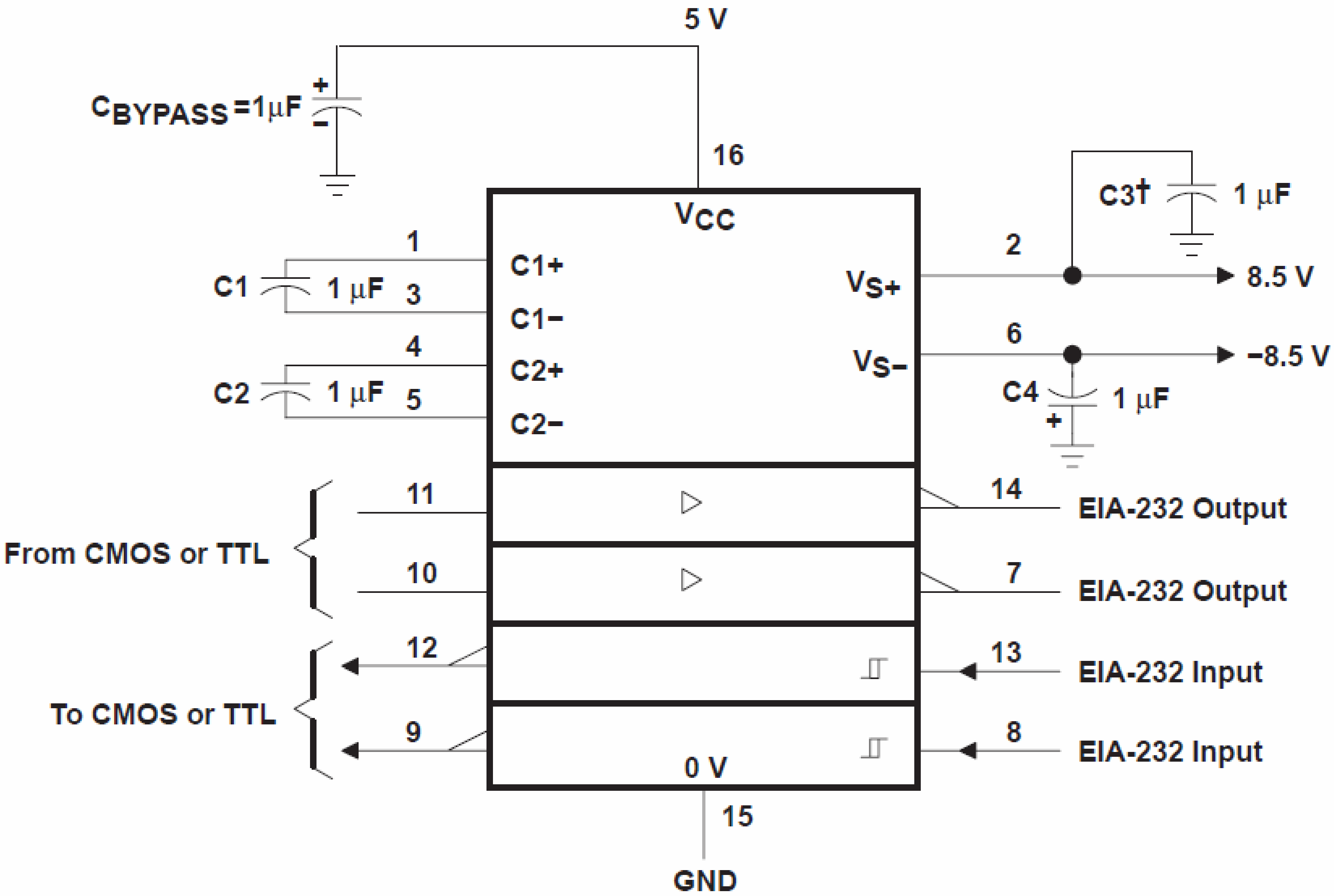
RS232C



Widok gniazda PC (męskiego) typu DE-9 od strony wtyczki

Numer		Kierunek	Oznaczenie	Nazwa angielska	Nazwa polska
9 pin	25 pin				
1	8	DCE – > DTE	DCD	Data Carrier Detected	sygnał wykrycia nośnej
2	3	DCE – > DTE	RxD	Receive Data	odbiór danych
3	2	DCE < – DTE	TxD	Transmit Data	transmisja danych
4	20	DCE < – DTE	DTR	Data Terminal Ready	gotowość terminala 1)
5	7	DCE – DTE	GND	Signal Ground	masa
6	6	DCE – > DTE	DSR	Data Set Ready	gotowość "modemu" 1)
7	4	DCE < – DTE	RTS	Request to Send Data	żądanie wysyłania
8	5	DCE – > DTE	CTS	Clear to Send Data	gotowość wysyłania
9	22	DCE – > DTE	RING	Ring indicator	wskaźnik dzwonka
	9-19; 21; 23-25		NC		nie wykorzystane 2)

MAX232



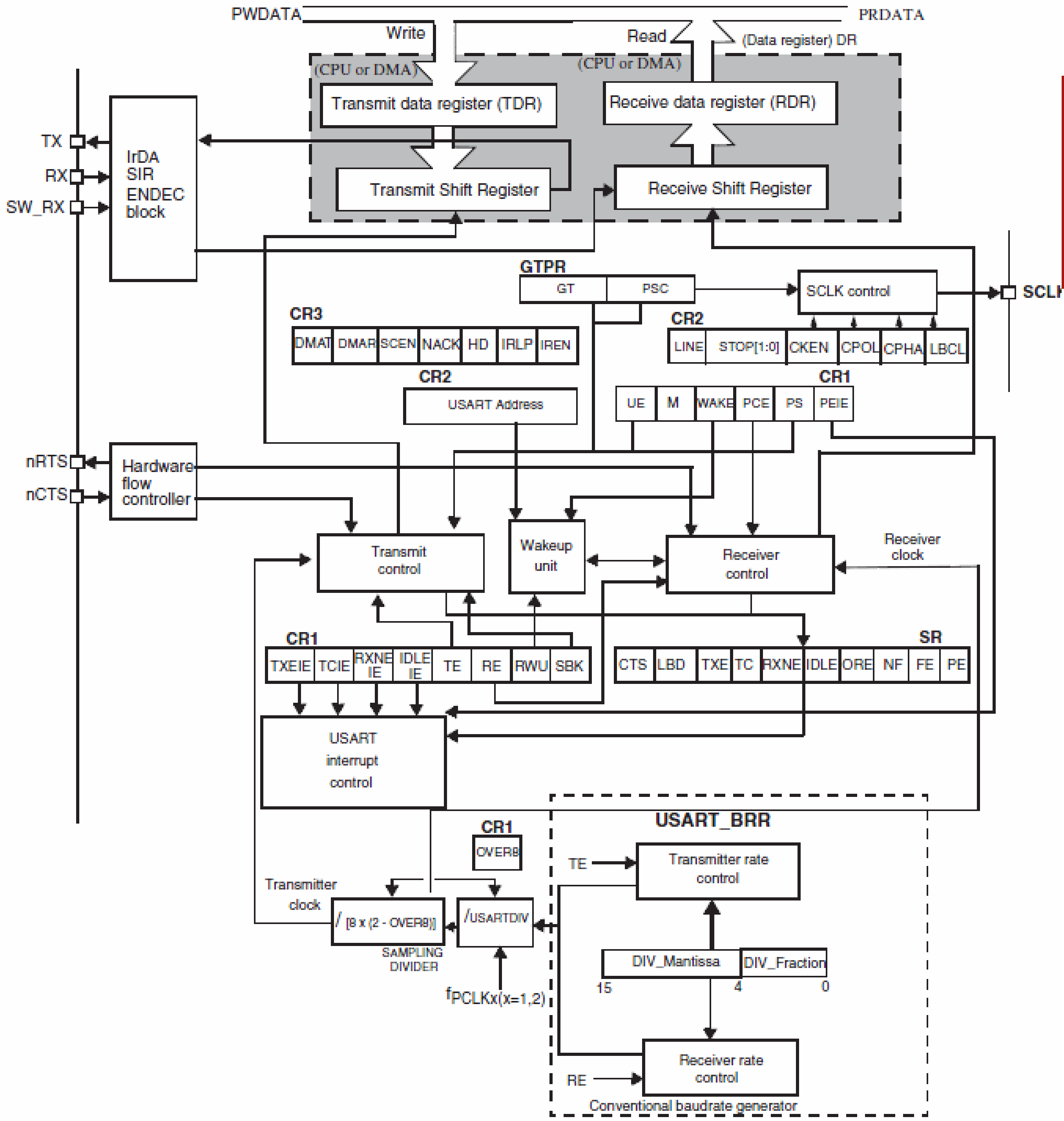
5 V
CBYPASS =1µF
+
–
16
VCC
C3†
1 µF
1
C1+
C1
1 µF
3
C1–
4
C2+
C2
1 µF
5
C2–
VS+
2
8.5 V
VS–
6
–8.5 V
C4
1 µF
+
From CMOS or TTL
11
10
14
EIA-232 Output
7
EIA-232 Output
To CMOS or TTL
12
9
13
EIA-232 Input
8
EIA-232 Input
0 V
15
GND

USART in STM32F4

- Features:
  - Full duplex, asynchronous or **synchronous** communications
  - Fractional baud rate generator systems - Common programmable transmit and receive baud rate
  - Programmable data word length (8 or 9 bits)
  - Configurable stop bits - support for 1 or 2 stop bits
  - Transmitter clock output for synchronous transmission
  - Single-wire half-duplex communication
  - Configurable multibuffer communication using DMA (direct memory access)
    - Buffering of received/transmitted bytes in reserved SRAM using centralized DMA

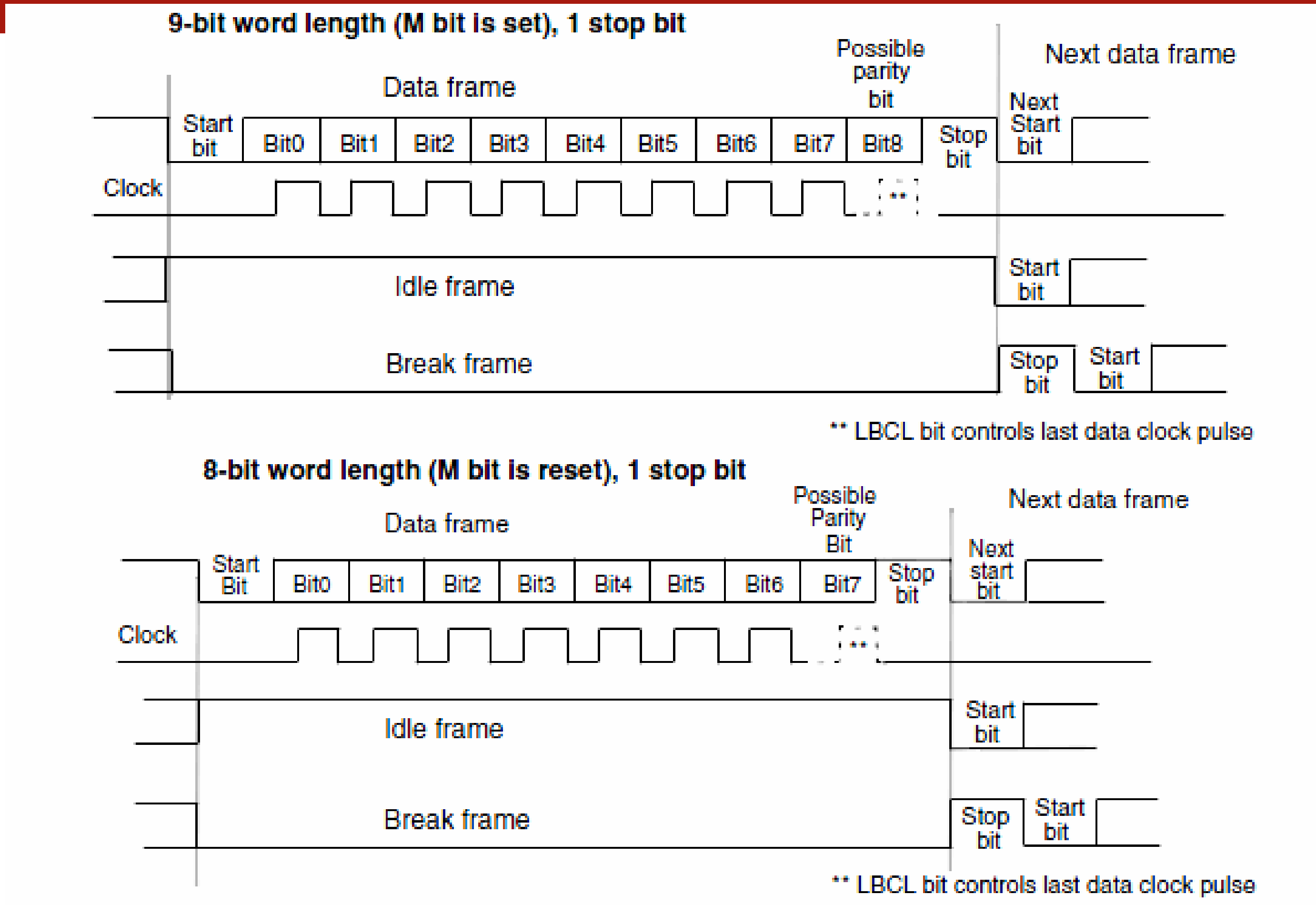
Poli

U



PWDATA
PRDATA
Write
Read
(Data register) DR
(CPU or DMA)
(CPU or DMA)
Transmit data register (TDR)
Receive data register (RDR)
Transmit Shift Register
Receive Shift Register
TX
RX
SW_RX
IrDA
SIR
ENDEC
block
GTPR
GT
PSC
SCLK control
SCLI
CR3
DMAT
DMAR
SCEN
NACK
HD
IRLP
IREN
CR2
LINE
STOP[1:0]
CKEN
CPOL
CPHA
LBCL
CR2
USART Address
CR1
UE
M
WAKE
PCE
PS
PEIE
nRTS
nCTS
Hardware
flow
controller
Transmit
control
Wakeup
unit
Receiver
control
Receiver
clock
CR1
TXEIE
TCIE
RXNE
IE
IDLE
IE
TE
RE
RWU
SBK
SR
CTS
LBD
TXE
TC
RXNE
IDLE
ORE
NF
FE
PE
USART
interrupt
control
CR1
OVER8
USART_BRR
TE
Transmitter rate
control
Transmitter
clock
/ [8 x (2 - OVER8)]
SAMPLING
DIVIDER
/USARTDIV
DIV_Mantissa
DIV_Fraction
15
4
0
fPCLKx(x=1,2)
Receiver rate
control
RE
Conventional baudrate generator

USART in STM32F4



USART in STM32F4

- For asynchronous mode only two pins necessary:
  - **RX:** Receive Data Input
  - **TX:** Transmit Data Output
- Additional pin necessary for synchrounous mode:
  - **SCLK:** Transmitter clock output. This pin outputs the transmitter data clock for synchronous transmission
- Pins required in Hardware flow control mode:
  - **nCTS:** Clear To Send blocks the data transmission at the end of the current transfer when high
  - **nRTS:** Request to send indicates that the USART is ready to receive a data (when low).

USART in STM32F4

- Important registers:
  – USART_SR – status register

Address offset: 0x00

Reset value: 0x00C0 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NF	FE	PE
						rc_w0	rc_w0	r	rc_w0	rc_w0	r	r	r	r	r

  – USART_DR – data register (8-bit value)

USART in STM32F4

– USART_BRR – baud rate register

Address offset: 0x08

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV_Mantissa[11:0]												DIV_Fraction[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

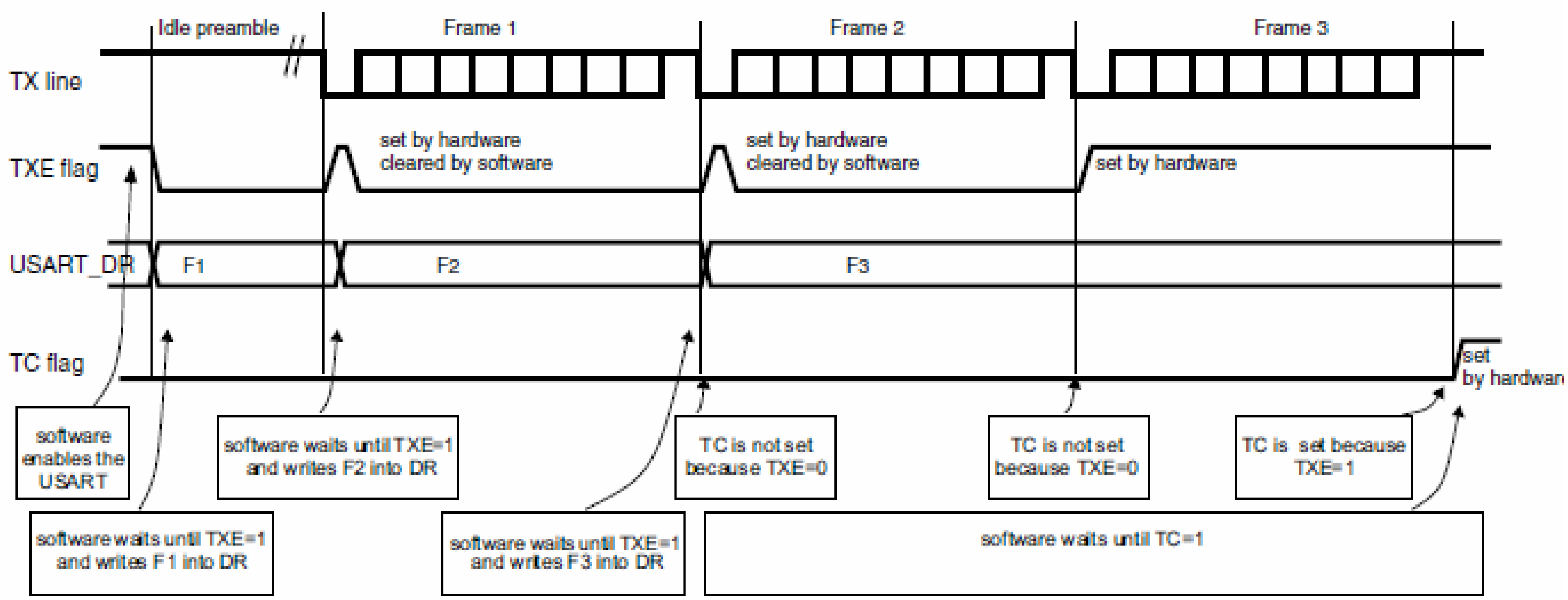
Oversampling by 16 (OVER8=0)							
Baud rate7		f_{PCLK} = 8 MHz			f_{PCLK} = 12 MHz		
S.No	Desired	Actual	Value programmed in the baud rate register	% Error = (Calculated - Desired) B.rate / Desired B.rate	Actual	Value programmed in the baud rate register	% Error
5	38.4 KBps	38.462 KBps	13	0.16	38.339 KBps	19.5625	0.16
6	57.6 KBps	57.554 KBps	8.6875	0.08	57.692 KBps	13	0.16
7	115.2 KBps	115.942 KBps	4.3125	0.64	115.385 KBps	6.5	0.16
8	230.4 KBps	228.571 KBps	2.1875	0.79	230.769 KBps	3.25	0.16
9	460.8 KBps	470.588 KBps	1.0625	2.12	461.538 KBps	1.625	0.16
10	921.6 KBps	NA	NA	NA	NA	NA	NA
11	2 MBps	NA	NA	NA	NA	NA	NA
12	3 MBps	NA	NA	NA	NA	NA	NA

USART in STM32F4

- Important registers:
  - USART_CR1 – control register
  - USART_CR2 – control register
  - USART_CR3 – control register

USART in STM32F4

- TC/TXE behaviour when transmitting

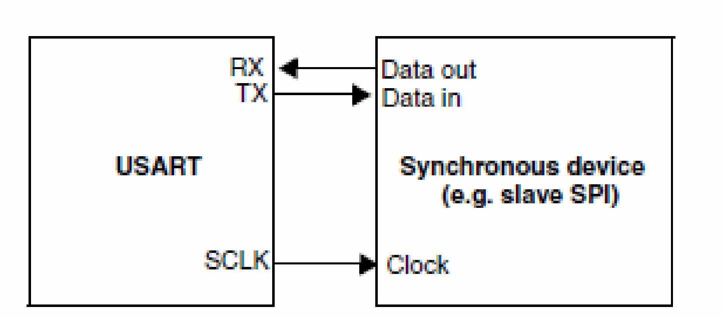


USART in STM32F4

- Parity control:
  - Parity control - generation of parity bit in transmission and parity checking in reception
  - Can be enabled by setting the PCE bit in the USART_CR1 register
  - Parity can be Even or Odd

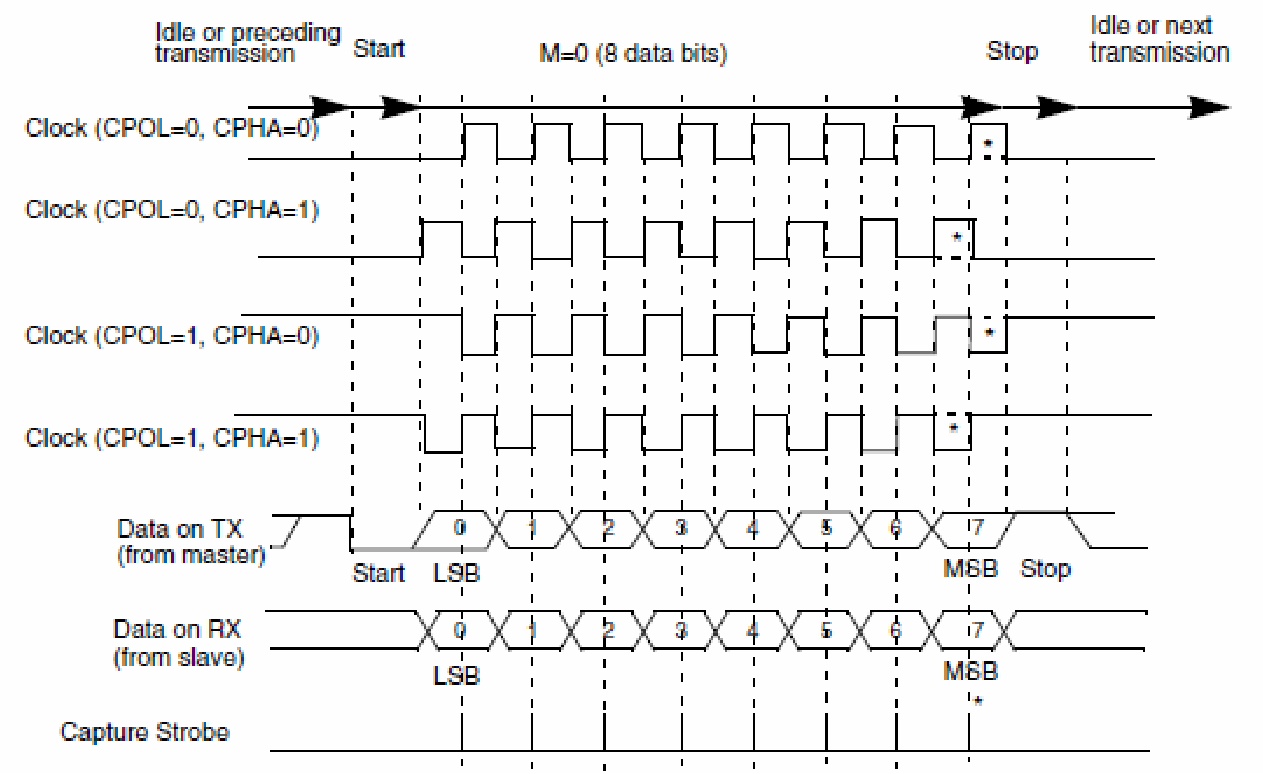
USART in STM32F4

- Synchronous mode:
  - Full duplex, clock controlled mode used for fast data transfer to **slave** devices



USART in STM32F4

- Synchronous mode:

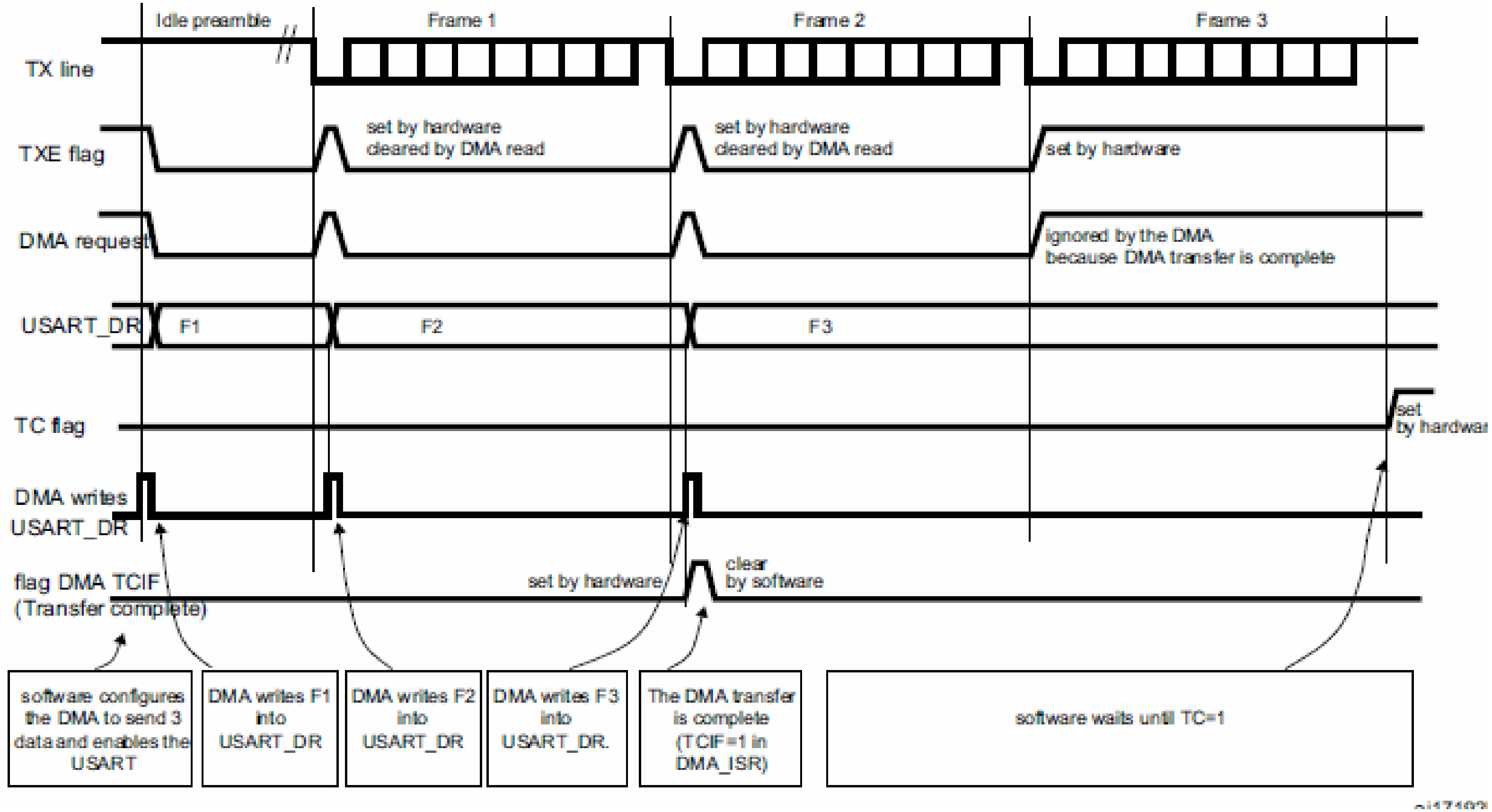


USART in STM32F4

- Half-duplex operation:
  - the TX and RX lines are internally connected
  - the RX pin is no longer used
  - the TX pin is always released when no data is transmitted. Thus, it acts as a standard I/O in idle or in reception.

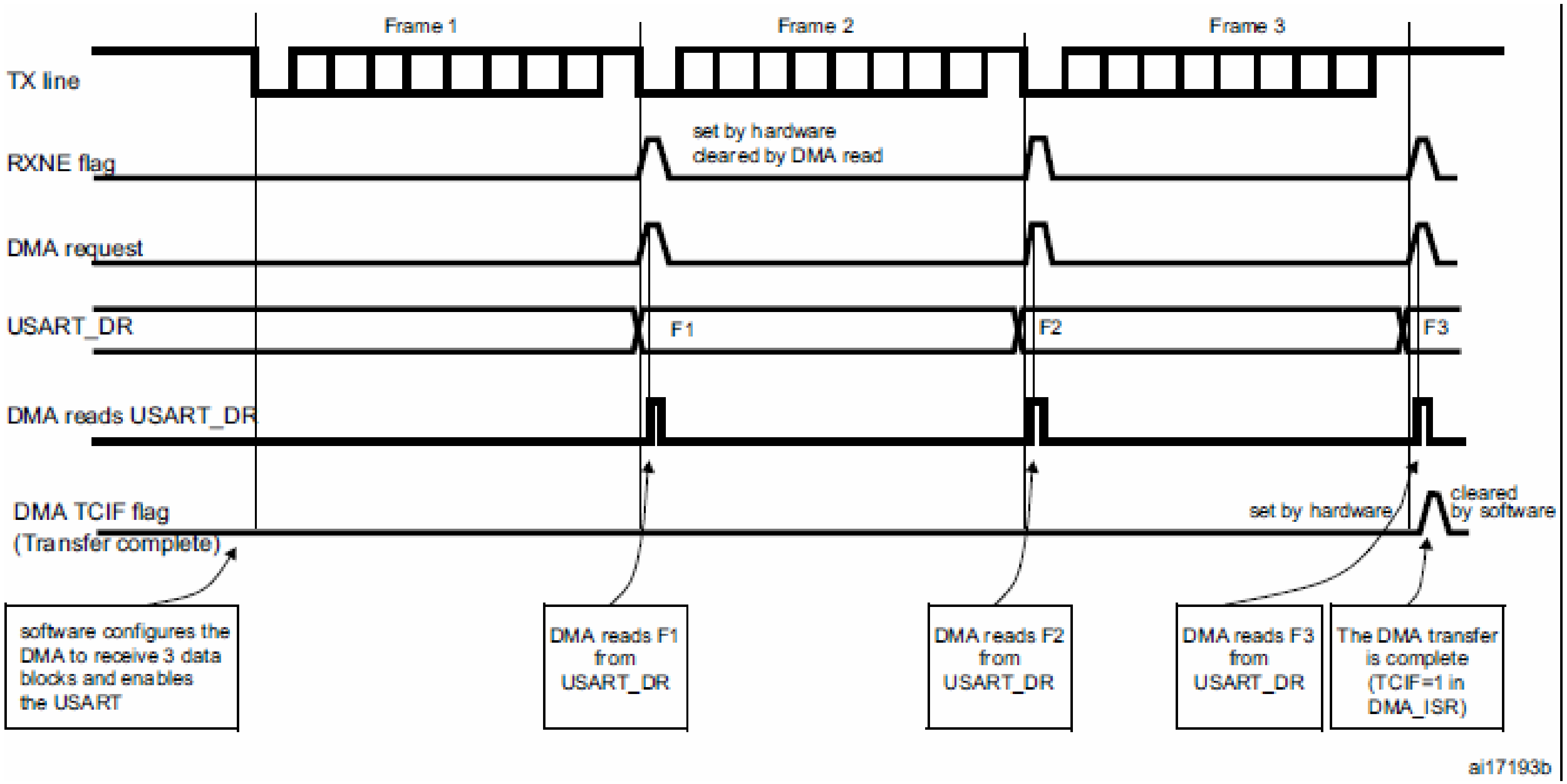
USART in STM32F4

- Continuous operation with DMA:
  - Transmission



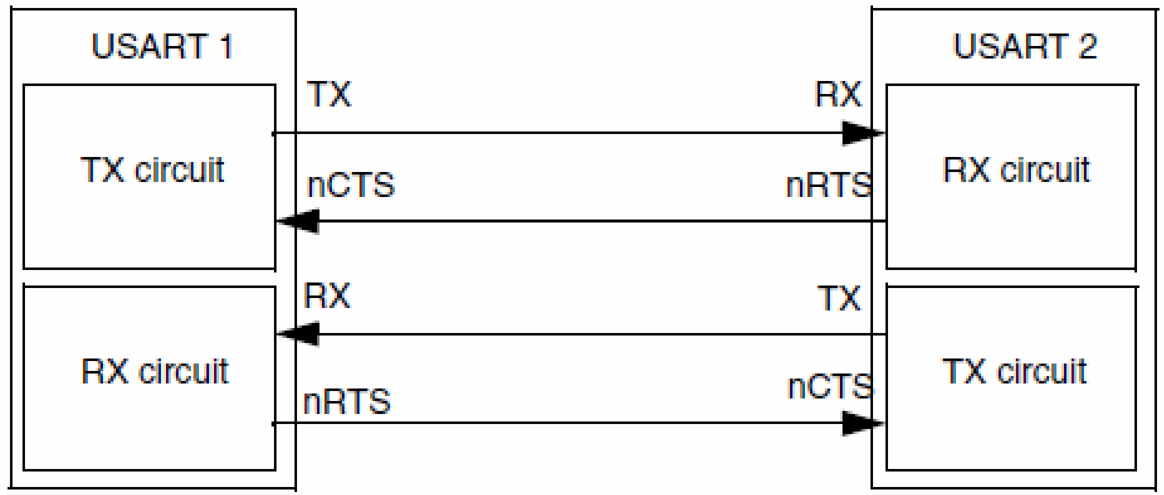
USART in STM32F4

- Continuous operation with DMA:
  - Reception



USART in STM32F4

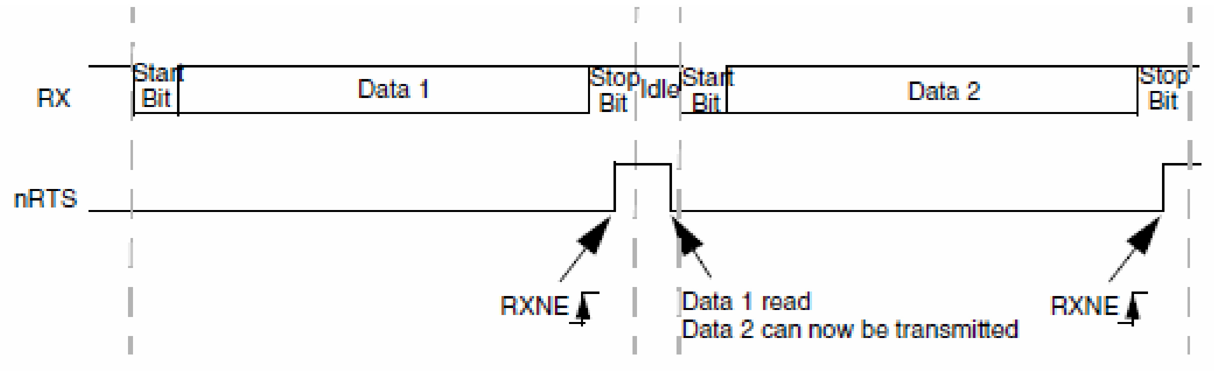
- Hardware flow control:
  - It is possible to control the serial data flow between 2 devices by using the nCTS input and the nRTS output



Politechnika Wrocławska

USART in STM32F4

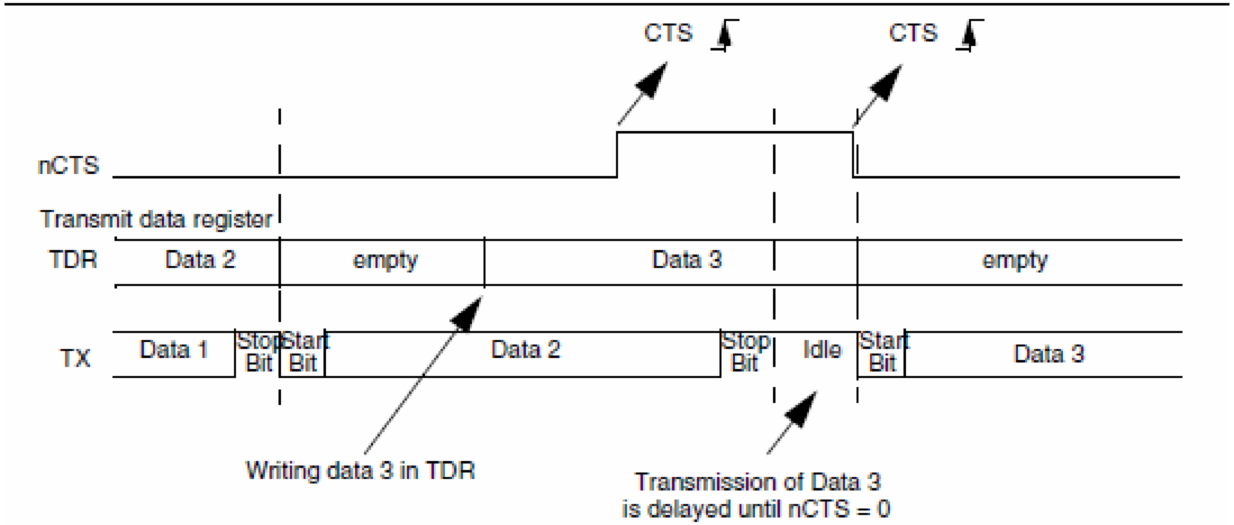
- Hardware flow control:
  - Request To Send - RTS flow control



  - nRTS is asserted (tied low) as long as the USART receiver is ready to receive a new data

USART in STM32F4

- Hardware flow control:
  - Clear To Send - CTS flow control



  - the transmitter checks the nCTS input before transmitting the next frame

USART in STM32F4

- Multiprocessor communication:
  - There is a possibility of performing multiprocessor communication with the USART (several USARTs connected in a network)
  - One of the USARTs is a master – its TX output is connected to the RX input of the other USARTs
  - The outputs of slaves are ANDed and connected to the RX of the master
  - Each processor has each own address (soft

Politechnika Wrocławska

USART in STM32F4

- USART Interrupts

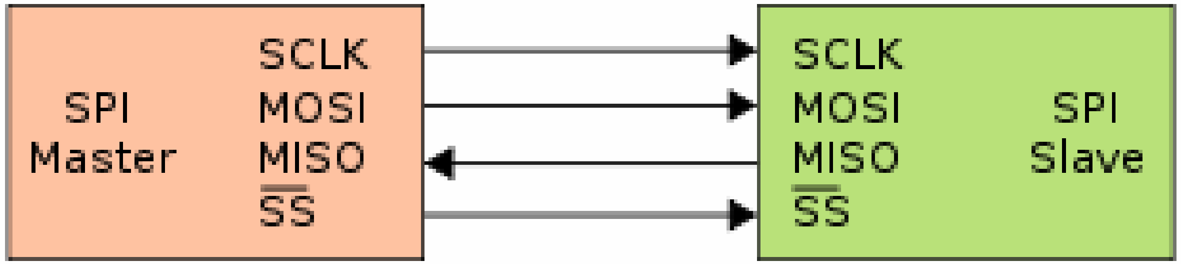
Interrupt event	Event flag	Enable control bit
Transmit Data Register Empty	TXE	TXEIE
CTS flag	CTS	CTSIE
Transmission Complete	TC	TCIE
Received Data Ready to be Read	RXNE	RXNEIE
Overrun Error Detected	ORE	RXNEIE
Idle Line Detected	IDLE	IDLEIE
Parity Error	PE	PEIE
Break Flag	LBD	LBDIE
Noise Flag, Overrun error and Framing Error in multibuffer communication	NF or ORE or FE	EIE

SPI Interface

SPI interface

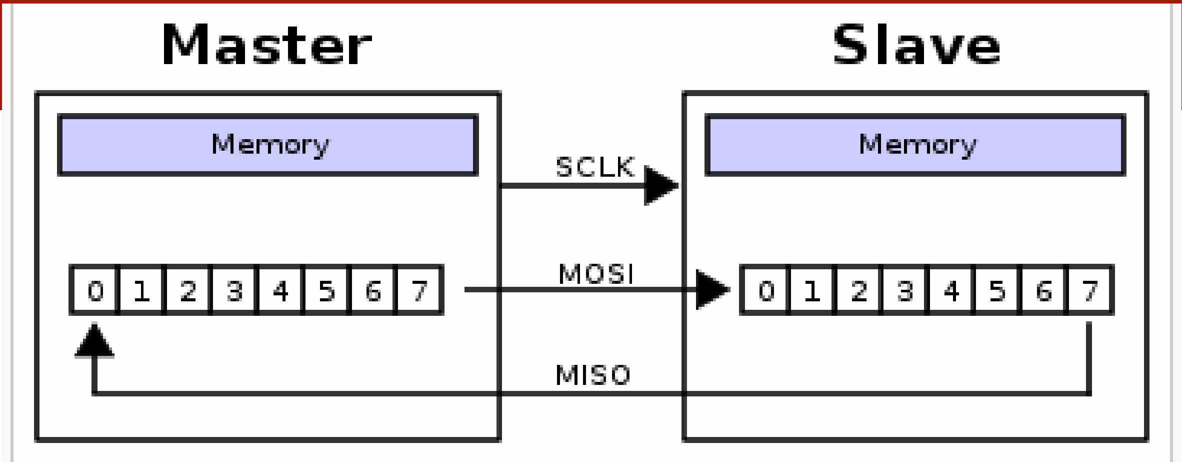
- Features:
  - One of the most popular serial interfaces
  - Serial, four-wire interface
  - Used for internal communication
  - Master-slave architecture
  - Two state TTL logic used!
  - Fully synchronous data transfer. Clock controlled by the master!
  - Full duplex transmission
  - Point – to – point transmission
  - Throughput up to tens of Mb/s

SPI - connection



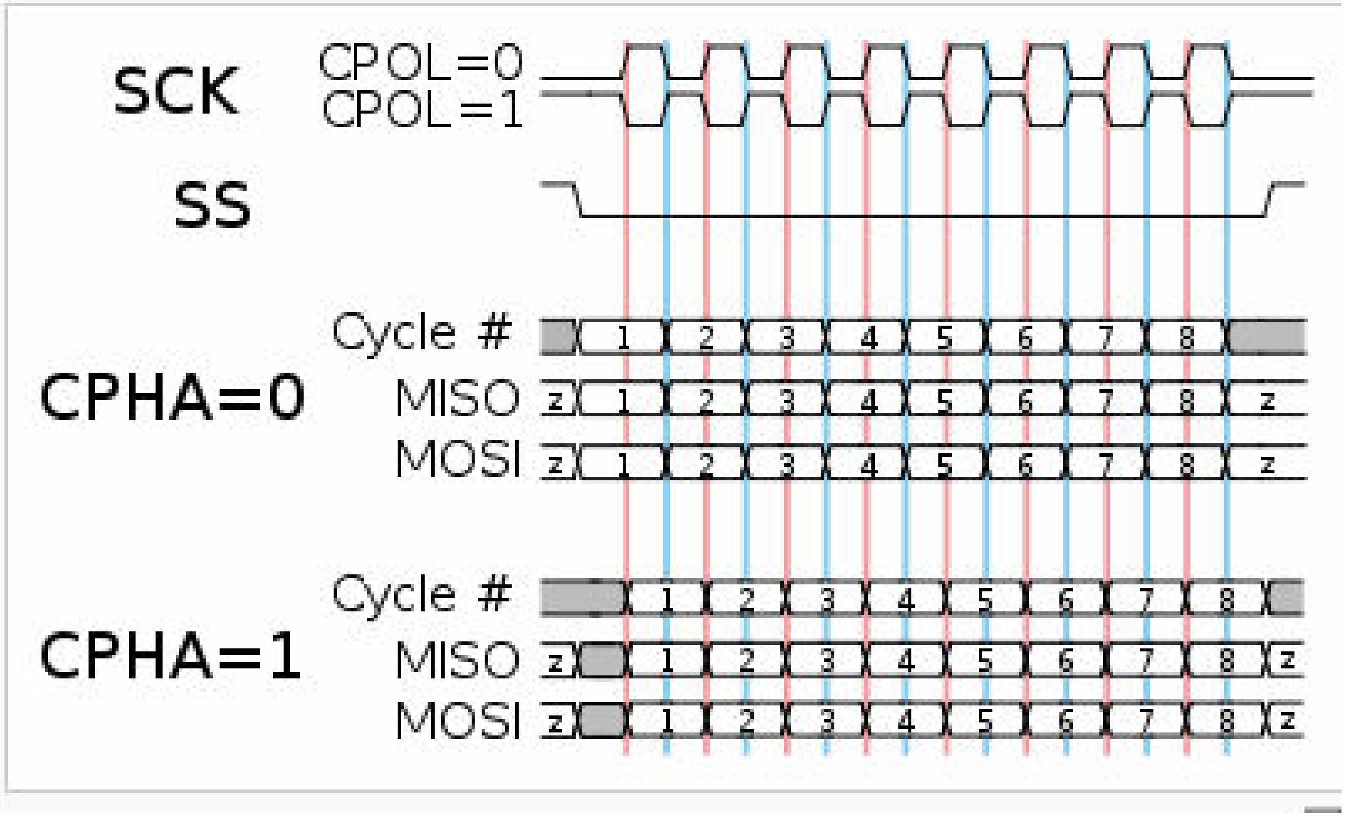
- *MISO – master input slave output*
- *MOSI – master output slave input*
- *SCLK – serial clock*
- *SS – slave select*

SPI - przebieg komunikacji



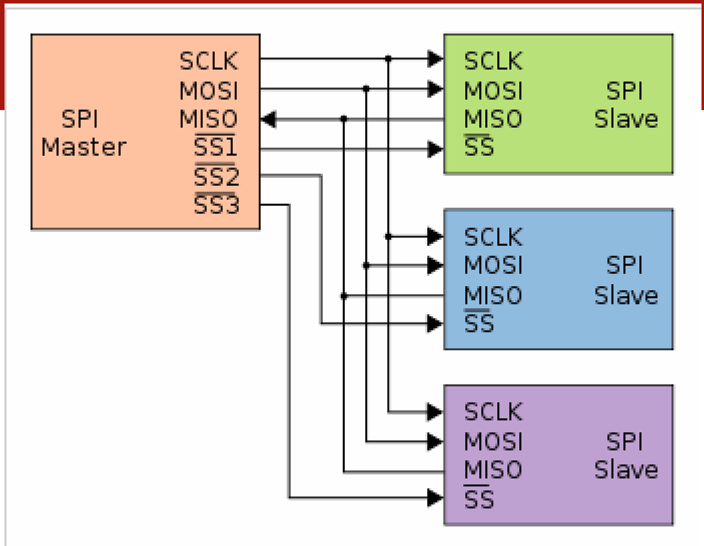
- *Both in Master and in Slave there are implemented shift registers (8-bits), SIPO and PISO type*
- *Zata are exchanged with the SCK signal*
- *After 8 (16) clock cycles the data transfer is finished*

SPI - configuration

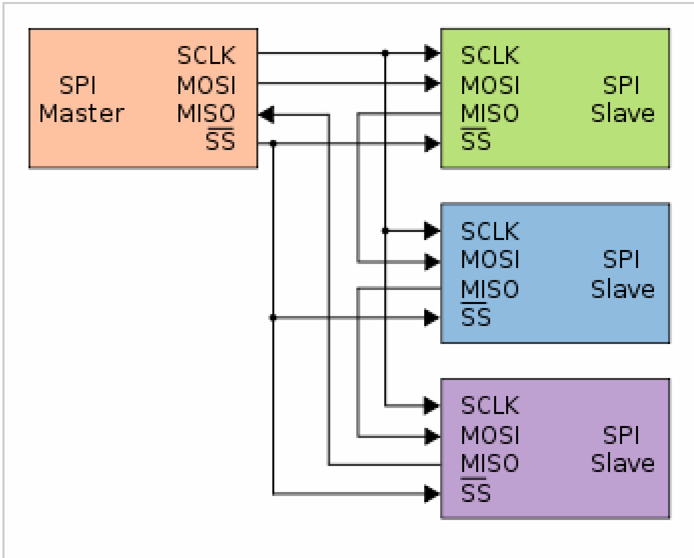


Politechnika Wrocławska

SPI - Multiple devices



- *Data bus*



- *Daisy chain*

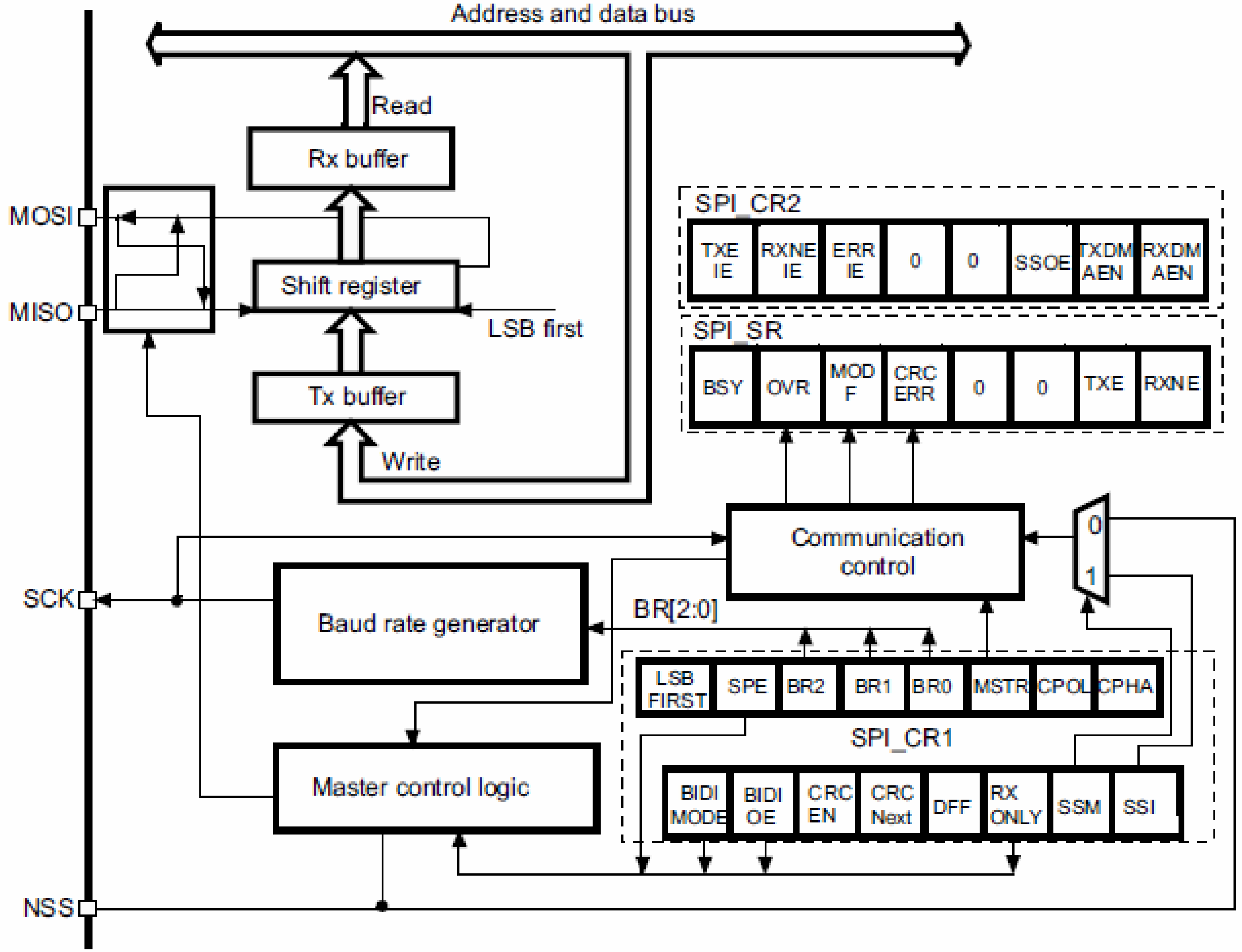
Politechnika Wrocławska

SPI in STM32F4

- Main features:
  - Full-duplex synchronous transfers on three lines
  - Simplex synchronous transfers on two lines with or without a bidirectional data line
  - 8- or 16-bit transfer frame format selection
  - Master or slave operation
  - Multimaster mode capability
  - Programmable clock polarity and phase
  - Programmable data order with MSB-first or LSB-first shifting
  - Hardware CRC feature for reliable communication
  - 1-byte transmission and reception buffer with DMA capability: Tx and Rx requests

Politechnika Wrocławska

SPI in STM32F4

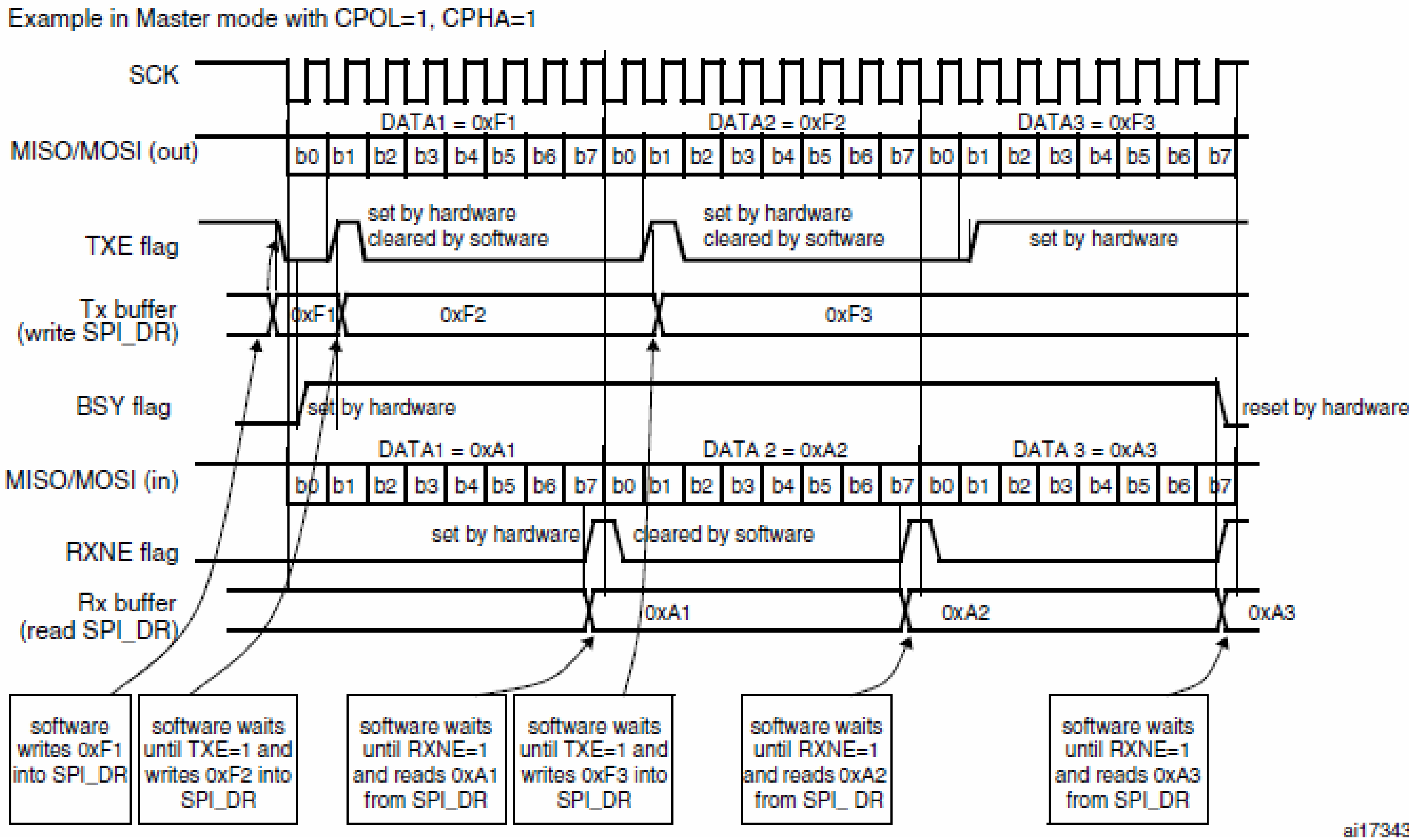


SPI in STM32F4

- Half-duplex operation
  - The SPI is capable of operating in half-duplex mode in 2 configurations:
    - 1 clock and 1 bidirectional data wire
    - 1 clock and 1 data wire (receive-only or transmit-only)

SPI in STM32F4

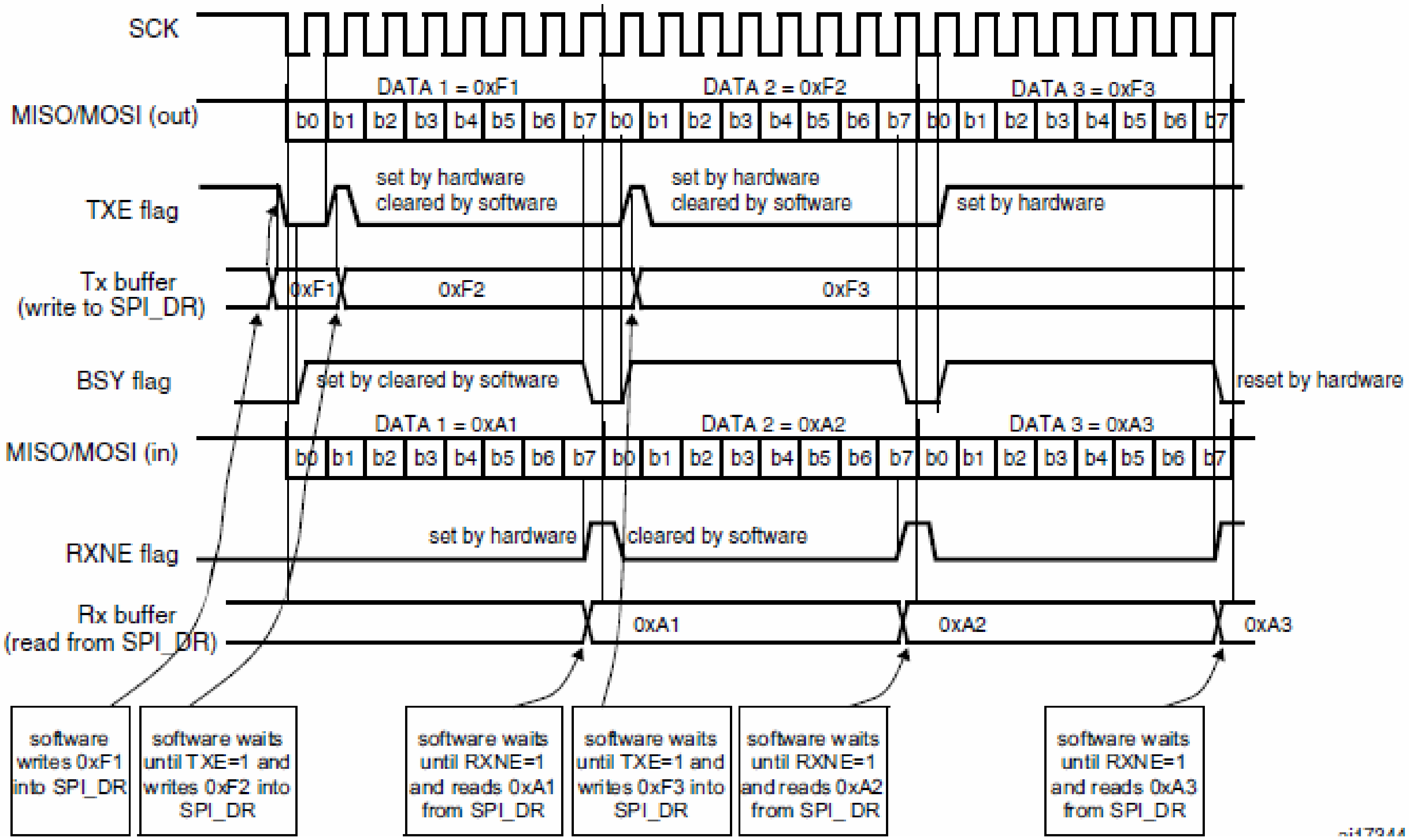
- Data transfer in full-duplex Master mode



SPI in STM32F4

- Data transfer in full-duplex Slave mode

Example in Slave mode with CPOL=1, CPHA=1



SPI in STM32F4

- Main registers:
  - SPI_DR – 16b data register split into 2 buffers - one for writing (Transmit Buffer) and another one for reading (Receive buffer)
  - SPI_SR – status register

Address offset: 0x08

Reset value: 0x0002

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							FRE	BSY	OVR	MODF	CRC ERR	UDR	CHSID E	TXE	RXNE
							r	r	r	r	rc_w0	r	r	r	r

SPI in STM32F4

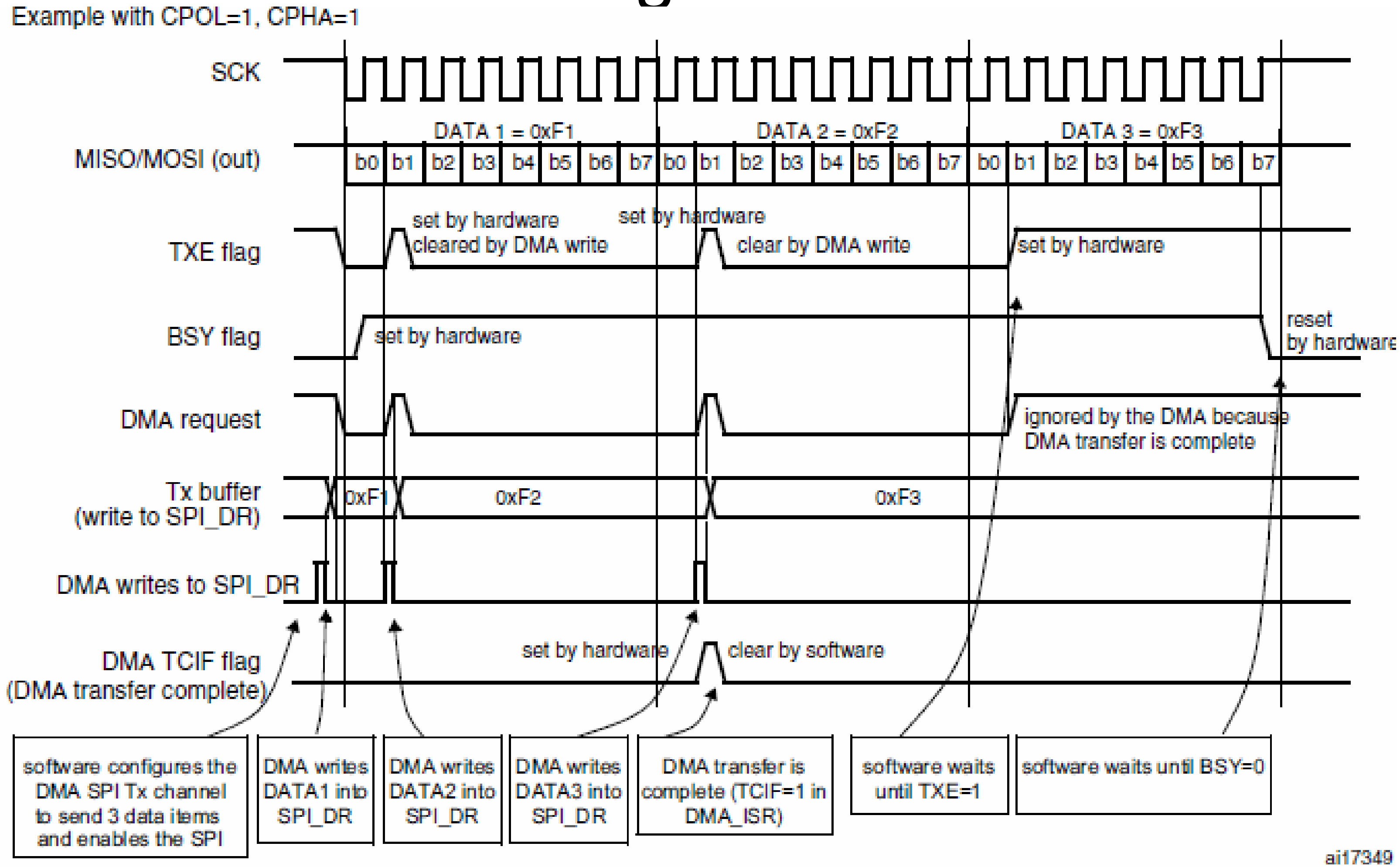
- Main registers:
  - SPI_CR1 – control register 1
  - SPI_CR2 – control register 2

SPI in STM32F4

- SPI communication using DMA:
  - SPI can be configure to operate at throughput exceeding 10Mb/s
  - To facilitate the transfers, the SPI features a DMA capability implementing a simple request/acknowledge protocol
  - In transmission, a DMA request is issued each time TXE is set to 1. The DMA then writes to the SPI_DR register.
  - In reception, a DMA request is issued each time RXNE is set to 1. The DMA then reads the SPI_DR register

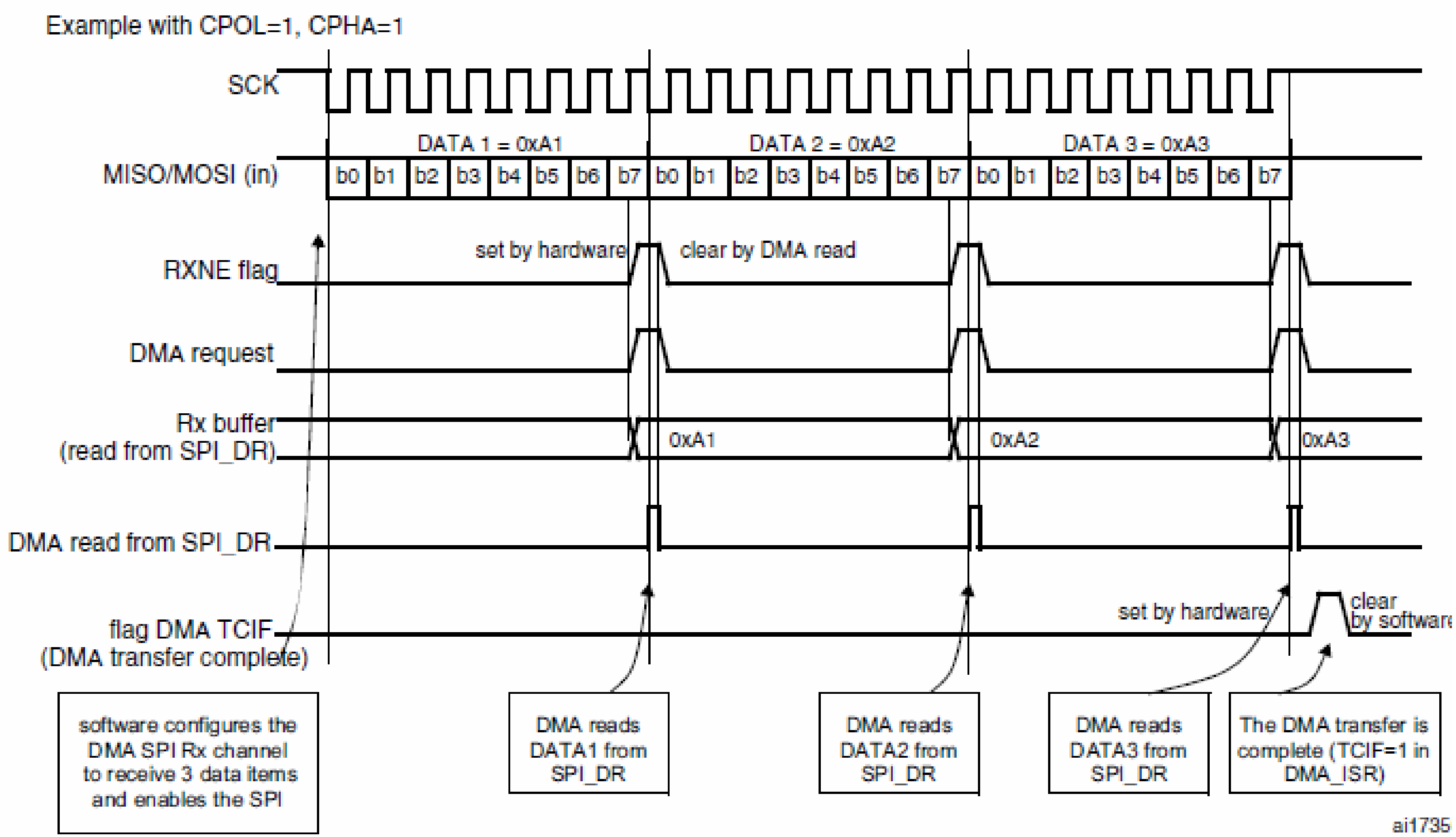
SPI in STM32F4

- Transmission using DMA



SPI in STM32F4

- Reception using DMA



SPI in STM32F4

- Interrupts:

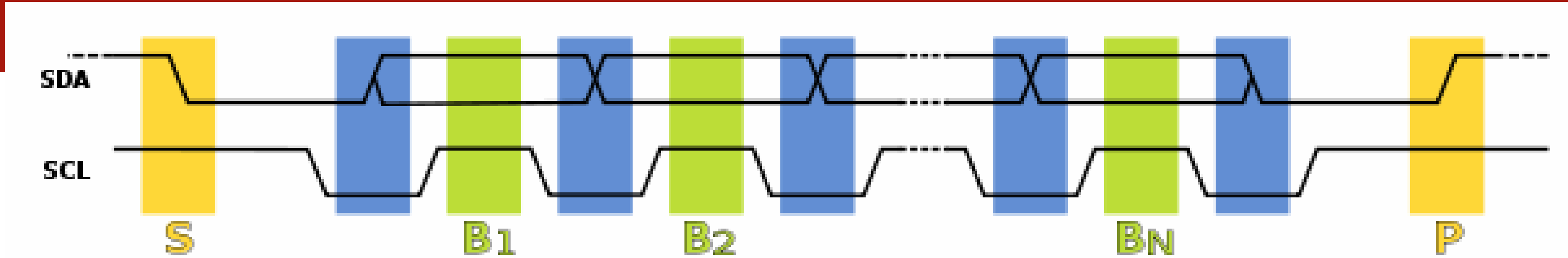
<table>
<tr><th>Interrupt event</th><th>Event flag</th><th>Enable Control bit</th></tr>
<tr><td>Transmit buffer empty flag</td><td>TXE</td><td>TXEIE</td></tr>
<tr><td>Receive buffer not empty flag</td><td>RXNE</td><td>RXNEIE</td></tr>
<tr><td>Master Mode fault event</td><td>MODF</td><td rowspan="3">ERRIE</td></tr>
<tr><td>Overrun error</td><td>OVR</td></tr>
<tr><td>CRC error flag</td><td>CRCERR</td></tr>
<tr><td>TI frame format error</td><td>FRE</td><td>ERRIE</td></tr>
</table>

I^2C Interface

I^2C interface

- Features:
  - Serial, two-wire interface
  - Used for internal communication
  - Master-slave architecture
  - Two state TTL logic used
  - Fully synchronous data transfer. Clock controlled by the master
  - Half-duplex transmission
  - Point – to – point transmission
  - Data rate 400kb/s (3.4Mb/s in HS mode)

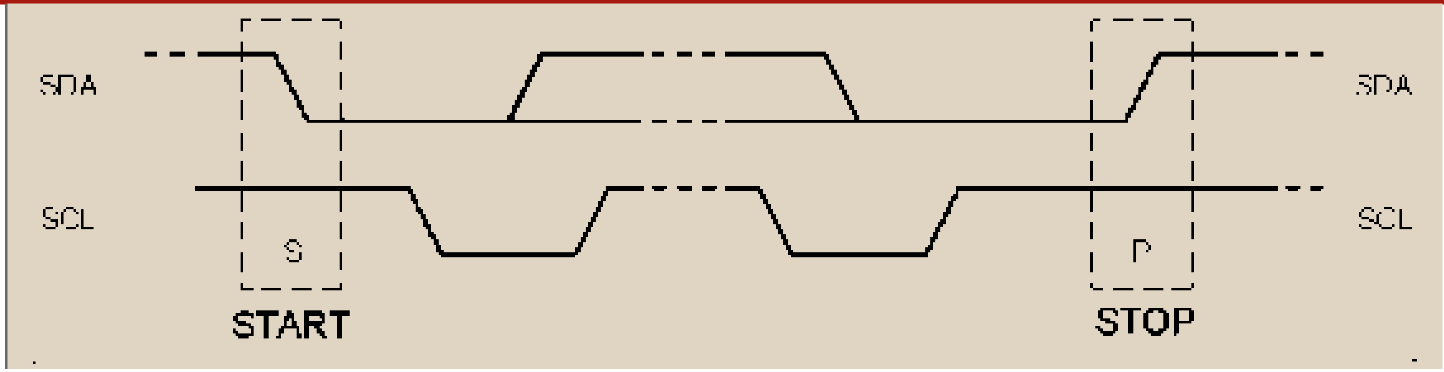
I²C - transmisja



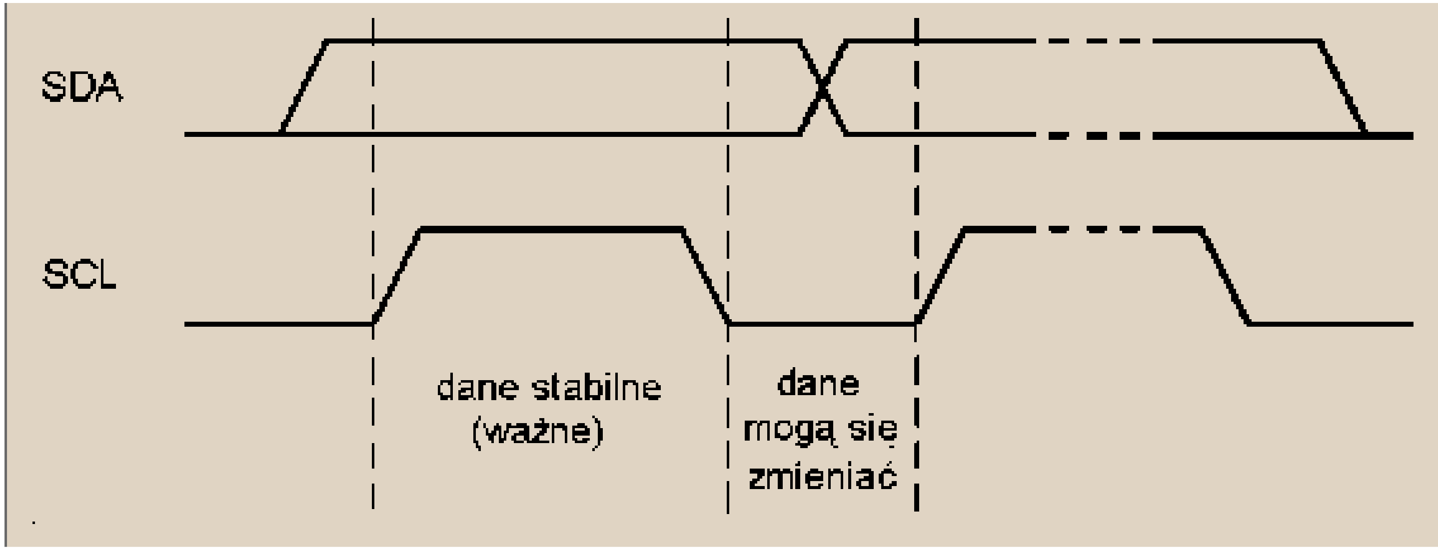
Features:

- *Transmission over two lines*
- *Both lines are open-drain type*
- *Both lines are pulled-up*
- *Each device checks, if there is no collision on the line*
- *Data transfer can be initiated only by the master*

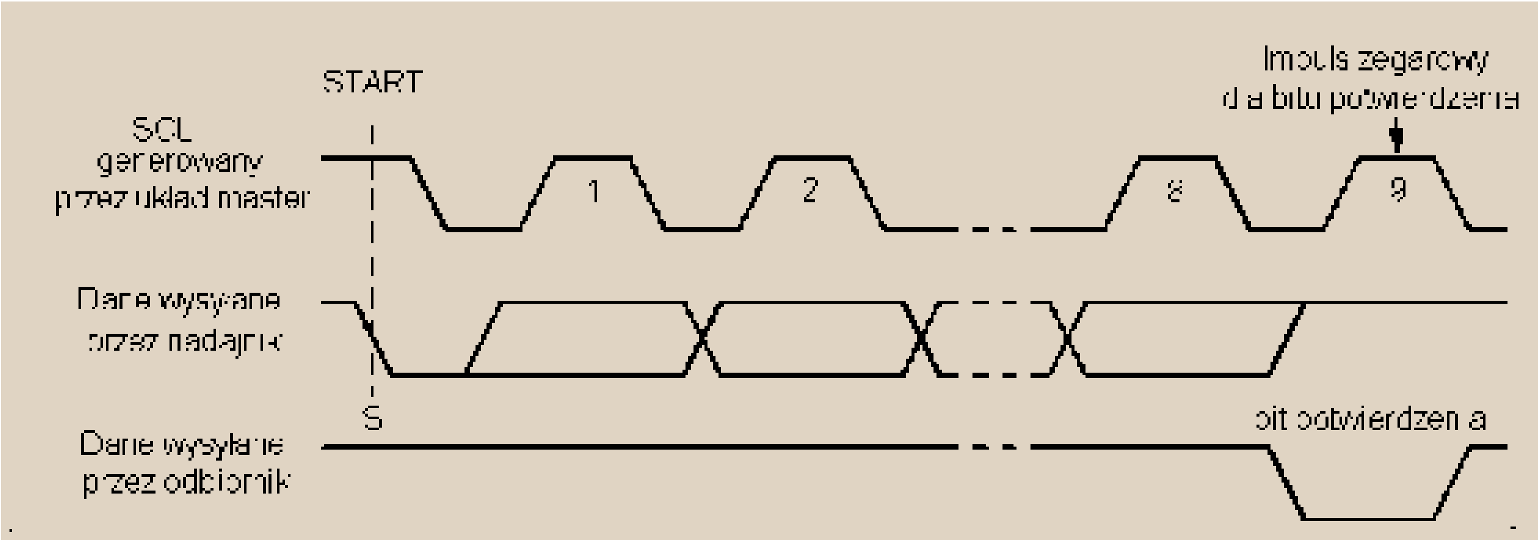
I²C -START/STOP



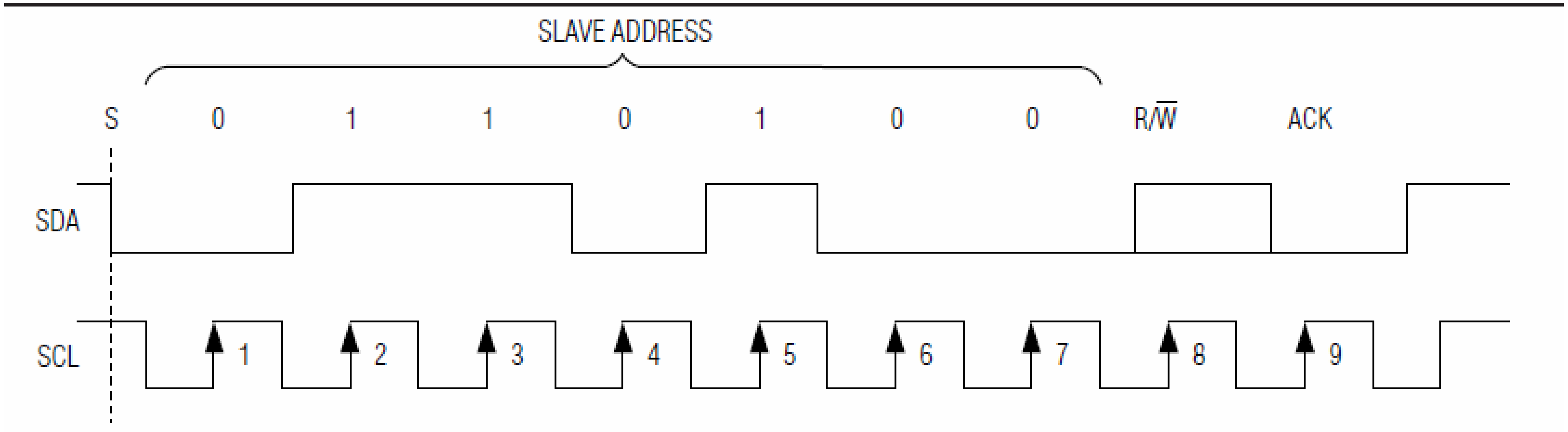
I²C - data transfer



I^{2}C -Acknowledge



I²C - addressing



- Each device has its own unique address
- Address is 7-bit long
- 8-th bit defines if there will be a read (1) or write (0) operation

I^2C - special modes

- In the first protocol specifications (1982) the max speed was set to 100 kb/s
- Nex the *Fast Mode* was introduced with the maximum speed of 400 kb/s
- In 2006 the *Fast Mode Plus* was defined with maximum speed of 1Mb/s
- With additional logic the, so called, *Highspeed Mode* can be implemented with maximum speed of 3.4 Mb/s

I^2C in STM32F4

- Main features:
  - Multimaster capability: the same interface can act as Master or Slave
  - I2C Master features:
    - Clock generation
    - Start and Stop generation
  - I2C Slave features:
    - Programmable I2C Address detection
    - Dual Addressing Capability to acknowledge 2 slave addresses
    - Stop bit detection

I^2C in STM32F4

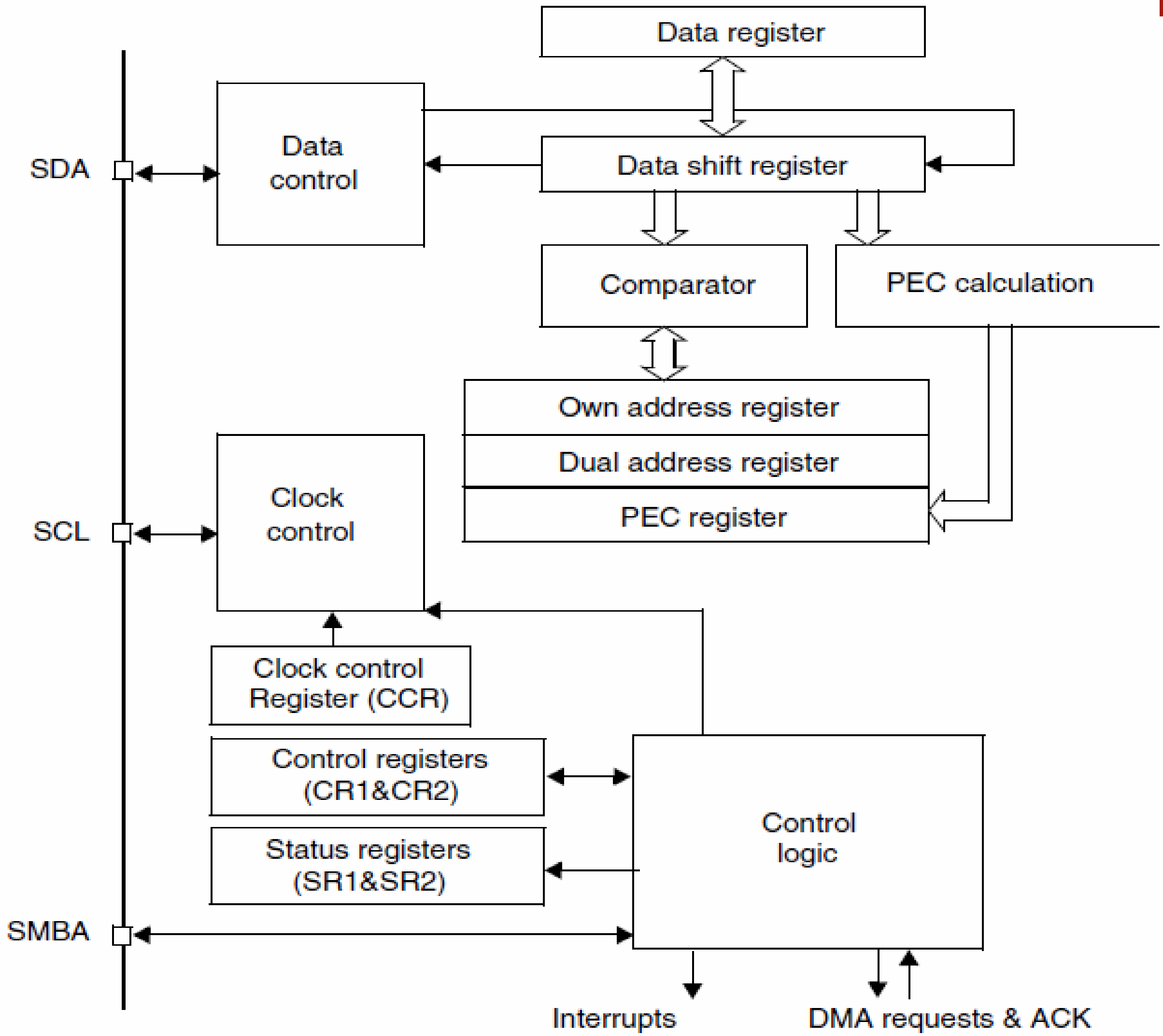
- Main features:
  - Generation and detection of 7-bit/10-bit addressing
  - Supports different communication speeds:
    - Standard Speed (up to 100 kHz)
    - Fast Speed (up to 400 kHz)
  - Programmable digital noise filter
  - Optional clock stretching
  - 1-byte buffer with DMA capability

I^2C in STM32F4

- Modes of operation:
  - Slave transmitter
  - Slave receiver
  - Master transmitter
  - Master receiver

  - By default the module operates in Slave mode
  - The interface automatically switches from slave to Master:
    - after it generates a START condition
  - The interface automatically switches from master to slave:
    - if an arbitration loss
    - a Stop generation occurs, allowing multimaster capability

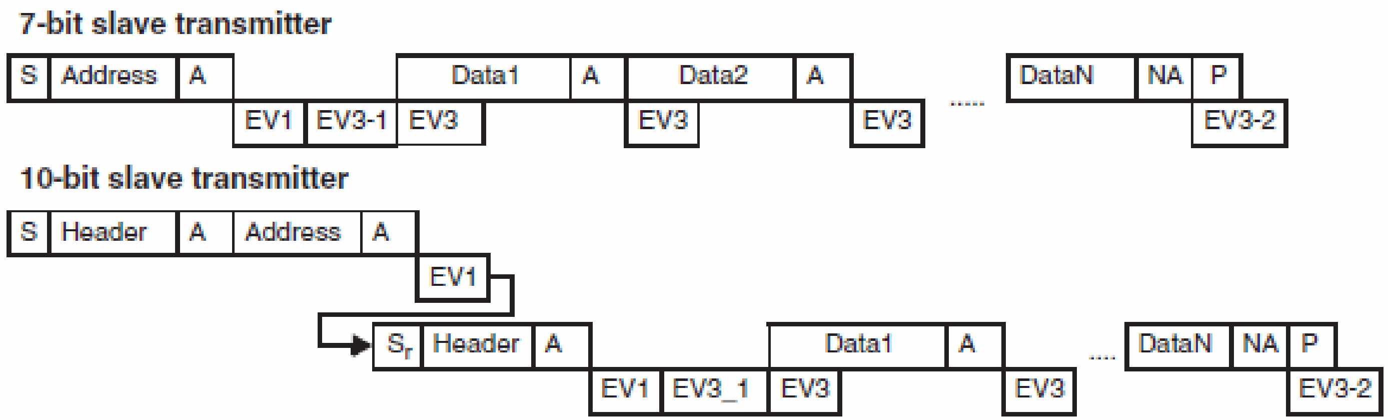
I²C in STM32F4



Politechnika Wrocławska

I^2C in STM32F4

- Slave transmitter

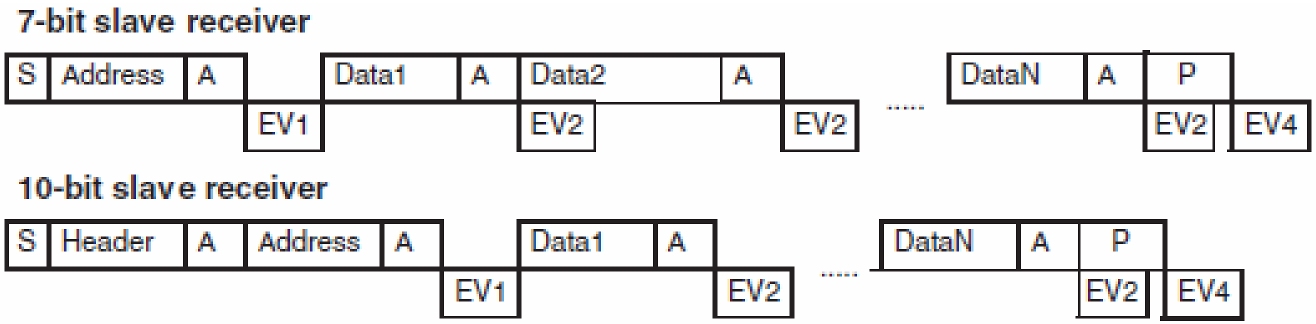


Legend: S= Start, S_r = Repeated Start, P= Stop, A= Acknowledge, NA= Non-acknowledge,
EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by reading SR2
EV3-1: TxE=1, shift register empty, data register empty, write Data1 in DR.
EV3: TxE=1, shift register not empty, data register empty, cleared by writing DR
EV3-2: AF=1; AF is cleared by writing '0' in AF bit of SR1 register.

I^{2}C in STM32F4

- Slave receiver

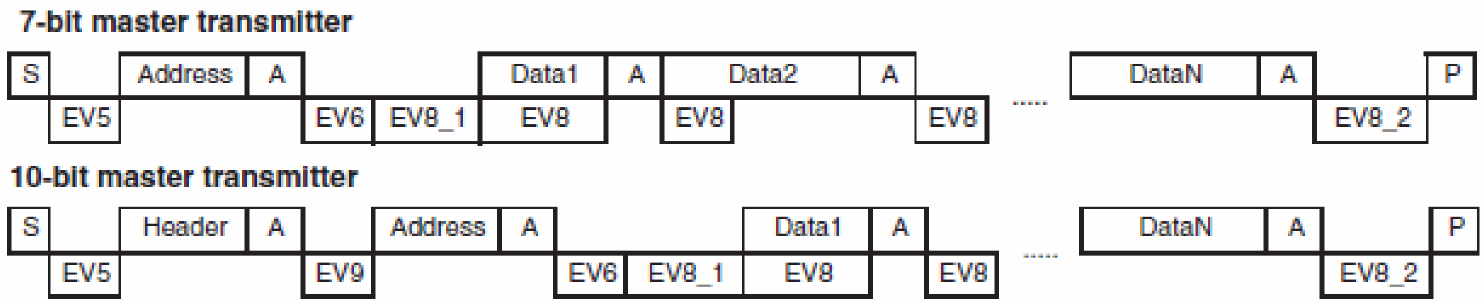


Legend: S= Start, S_r = Repeated Start, P= Stop, A= Acknowledge,
EVx= Event (with interrupt if ITEVFEN=1)

EV1: ADDR=1, cleared by reading SR1 followed by reading SR2
EV2: RxNE=1 cleared by reading DR register.
EV4: STOPF=1, cleared by reading SR1 register followed by writing to the CR1 register

I²C in STM32F4

- Master transmitter

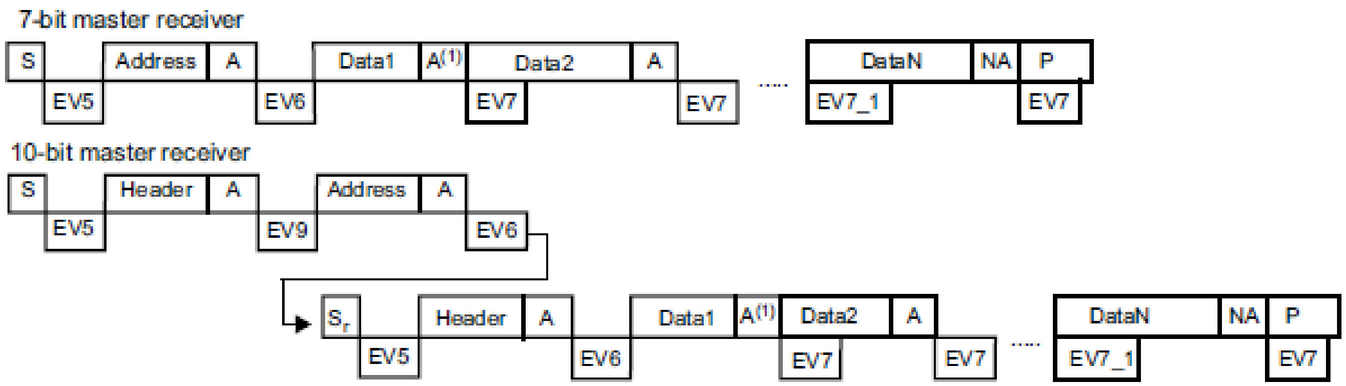


Legend: S= Start, S_r = Repeated Start, P= Stop, A= Acknowledge,
EVx= Event (with interrupt if ITEVFEN = 1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register with Address.
EV6: ADDR=1, cleared by reading SR1 register followed by reading SR2.
EV8_1: TxE=1, shift register empty, data register empty, write Data1 in DR.
EV8: TxE=1, shift register not empty,.data register empty, cleared by writing DR register
EV8_2: TxE=1, BTF = 1, Program Stop request. TxE and BTF are cleared by hardware by the Stop condition
EV9: ADD10=1, cleared by reading SR1 register followed by writing DR register.

I^{2}C in STM32F4

- Master receiver



Legend: S= Start, S_r = repeated Start, P = Stop, A= Ackowledge, NA = Non-acknowledge,
EVx= Event (with interrupt if ITEVFEN=1)

EV5: SB=1, cleared by reading SR1 register followed by writing DR register.

EV6: ADDR=1, cleared by reading SR1 register followed by reading SR2. In 10-bit master receiver mode, this sequence should be followed by writing CR2 with SART = 1.
In case of the reception of 1 byte, the Acknowledge disable must be performed suring EV6 event, i.e. before clearing ADDR flag.
EV7: RxNE = 1 cleared by reading DR register.
EV7_1: RxNE = 1 cleared by reading DR register, programming ACK = 0 and STOP request.
EV9: ADD10 = 1, cleared by reading SR1 register followed by writng DR register.

I^2C in STM32F4

- DMA:
  - DMA requests (when enabled) are generated only for data transfer
  - DMA requests are generated:
    - by Data Register becoming empty in transmission
    - Data Register becoming full in reception

I²C in STM32F4

- Interrupts:

Interrupt event	Event flag	Enable control bit
Start bit sent (Master)	SB	ITEVFEN
Address sent (Master) or Address matched (Slave)	ADDR	
10-bit header sent (Master)	ADD10	
Stop received (Slave)	STOPF	
Data byte transfer finished	BTF	
Receive buffer not empty	RxNE	ITEVFEN and ITBUFEN
Transmit buffer empty	TxE	
Bus error	BERR	ITERREN
Arbitration loss (Master)	ARLO	
Acknowledge failure	AF	
Overrun/Underrun	OVR	
PEC error	PECERR	
Timeout/Tlow error	TIMEOUT	
SMBus Alert	SMBALERT	

I^2C in STM32F4

- Main registers:
  - I2C_CCR – Clock control register
  - I2C_DR – data register
  - I2C_OAR1 – Own address register
  - I2C_SR1 – status register 1
  - I2C_SR2 – status register 2
  - I2C_CR1 – control register 1
  - I2C_CR2 – control register 2

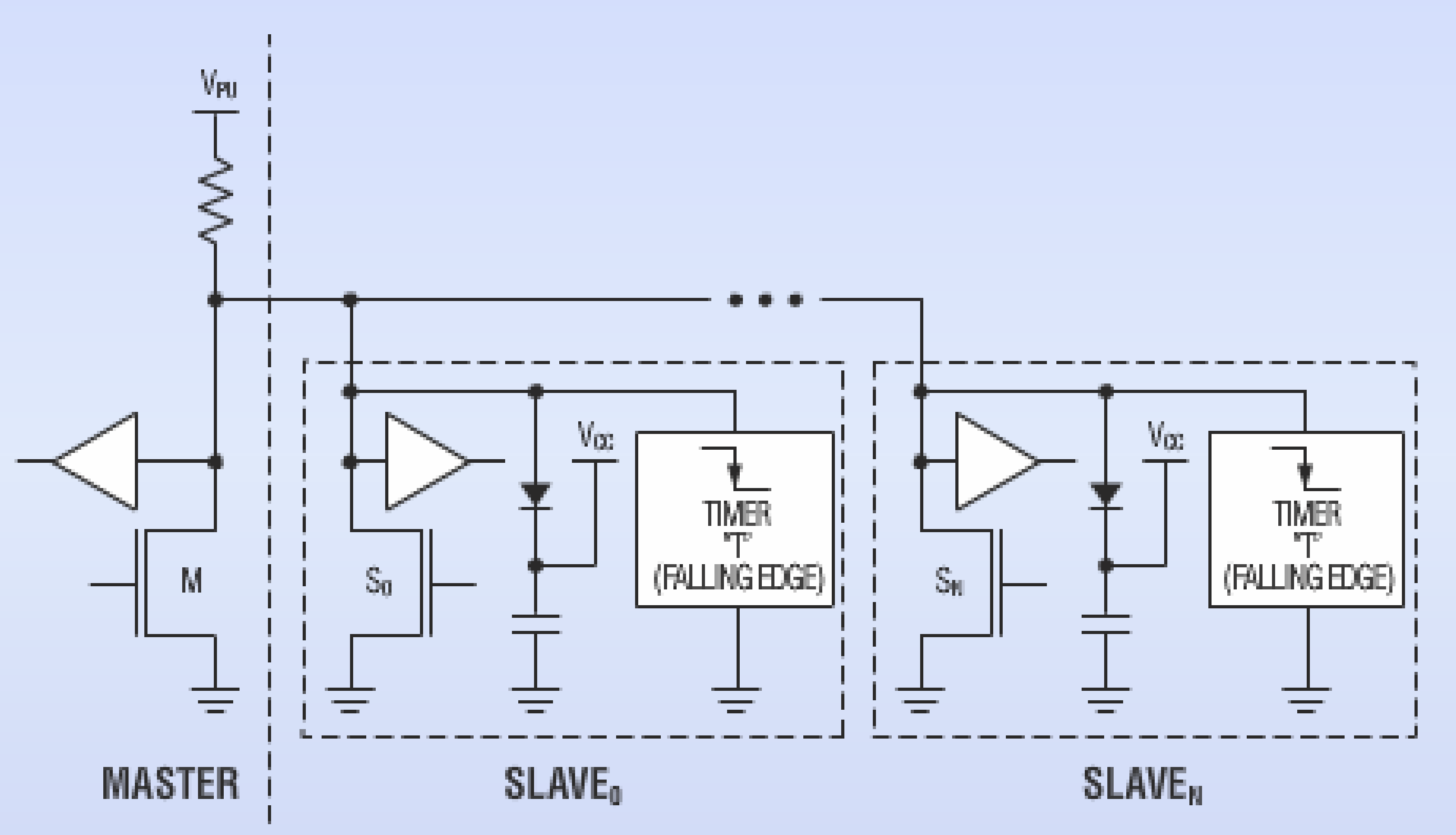
Politechnika Wrocławska

OneWire Interface

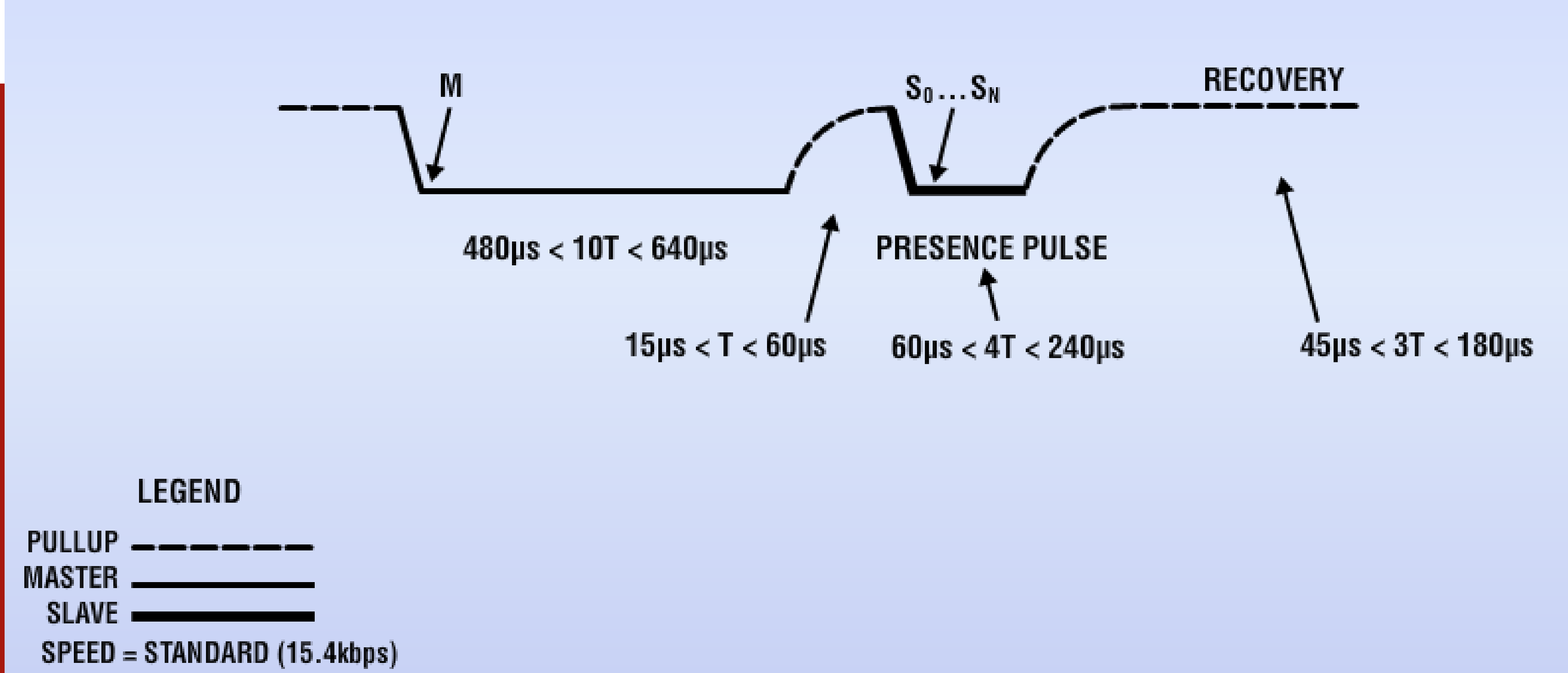
OneWire interface

- Features:
  - Serial, one-wire interface
  - Communcation wire can aalso be used for delivering power supply!
  - Used for internal communication but on longer distances
  - Master-slave architecture
  - Asynchronous data transfer. Data transfer controlled by the master
  - Half-duplex transmission
  - Point – to – point transmission
  - Data rate 15.4kb/s (standard) or 125kb/s (overdrive)

OneWire – podłączenie master/slave



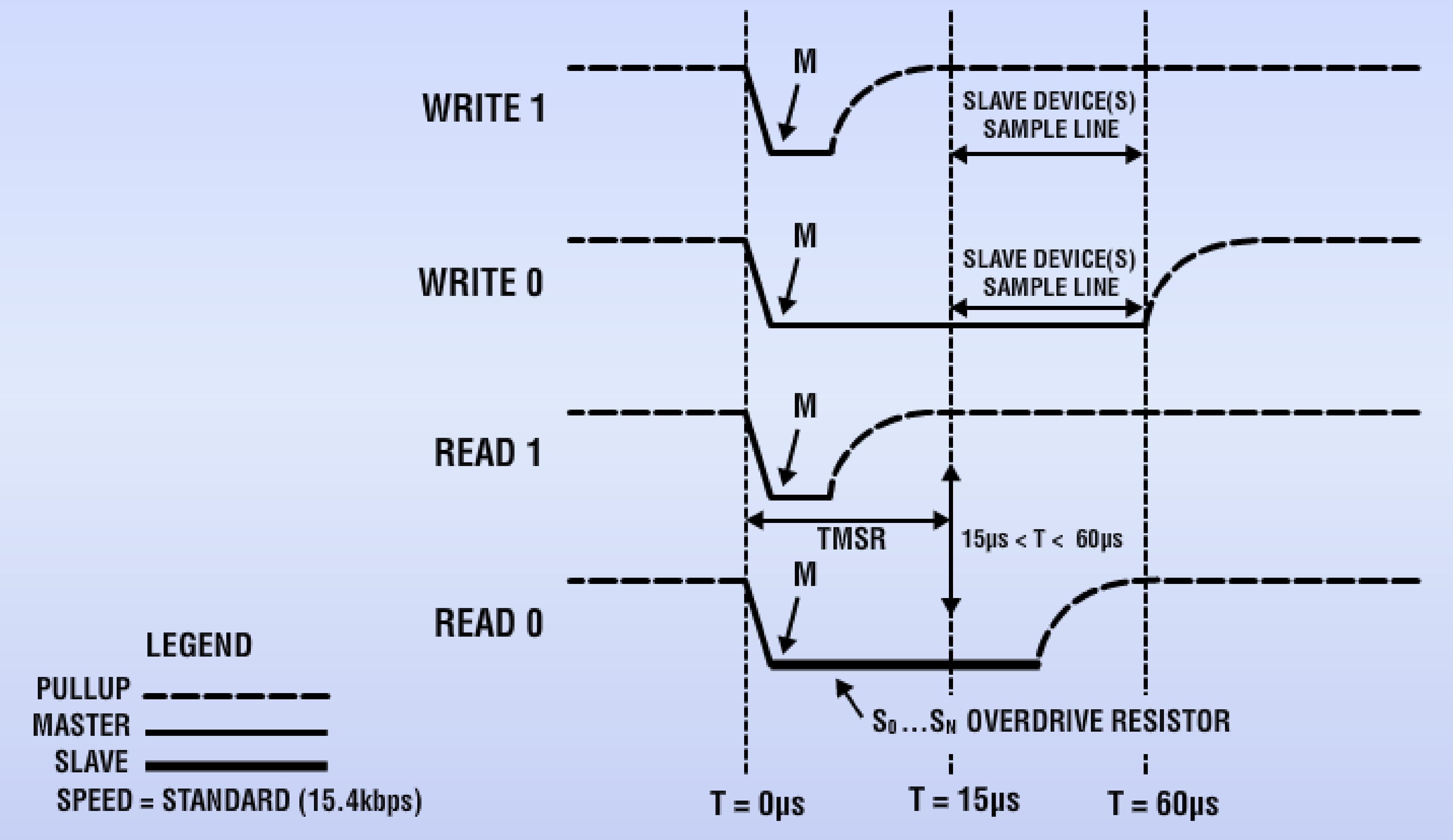
OneWire – reset/wykrywanie obecności



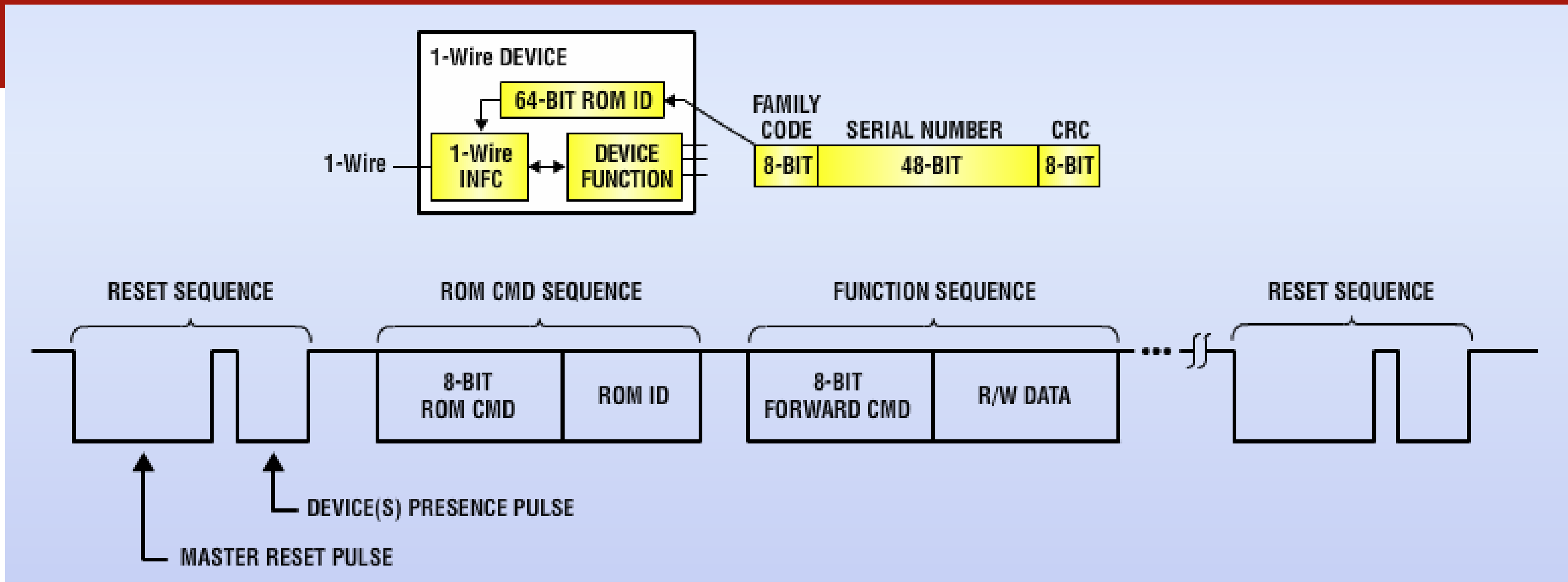
- *Precence detection*

Politechnika Wrocławska

OneWire - zapis/ odczyt

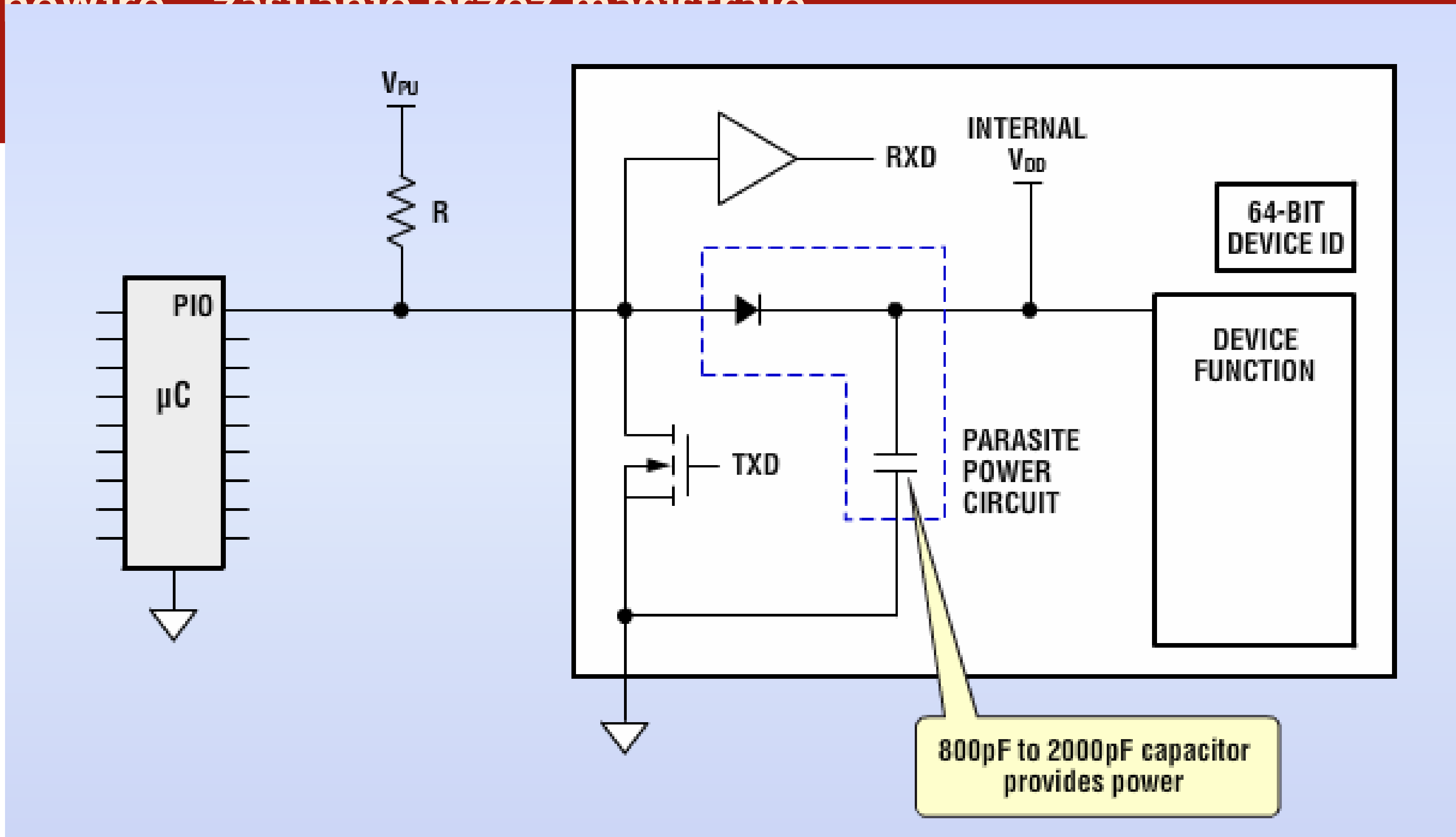


OneWire - fazy komunikacji



- *Communication has three main phases: RESET, authorisation (ROM commands), data read/write*
- *Each OneWire device has a unique 64-bit security code*

OneWire – zasilanie przez magistralę



- *Power supply delivery*

I^2S Interface

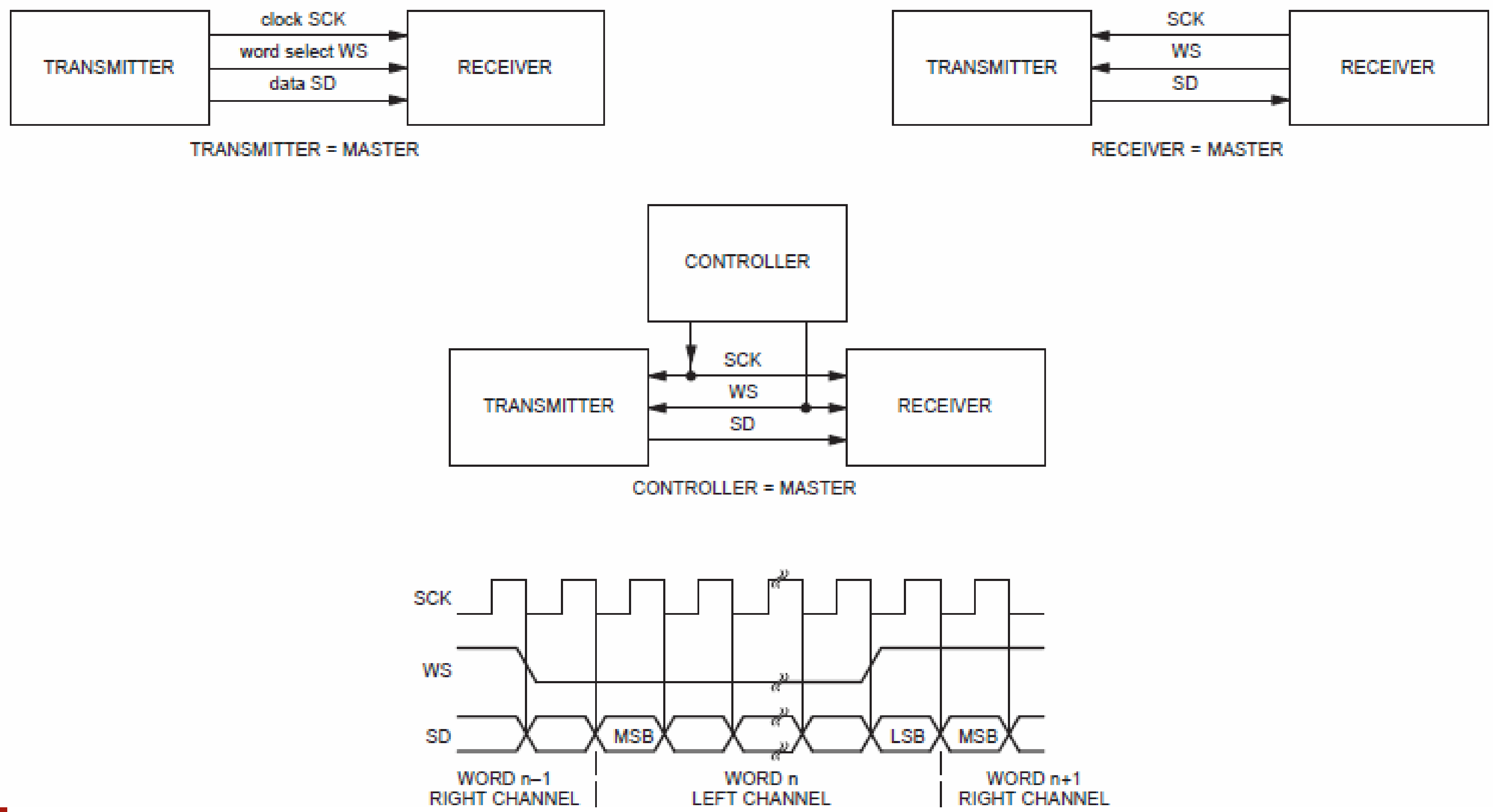
I²S interface

- Features:
  - Serial, bidirectional bus used to transfer data in electronic devices
  - Suitable for connection of digital audio
  - Developed by Philips (like I2C)
  - Also known abbreviation IIS
  - Separated clock and data bus (similar to SPI)
  - Transmission is via at least three lines SCK (clock), WS (select dates), SD (data) and ground GND
  - Speed depends on the type of data being transferred
  - Maximum speed exceeds 2 Mb / s (eg 32b data at 44.1kHz)

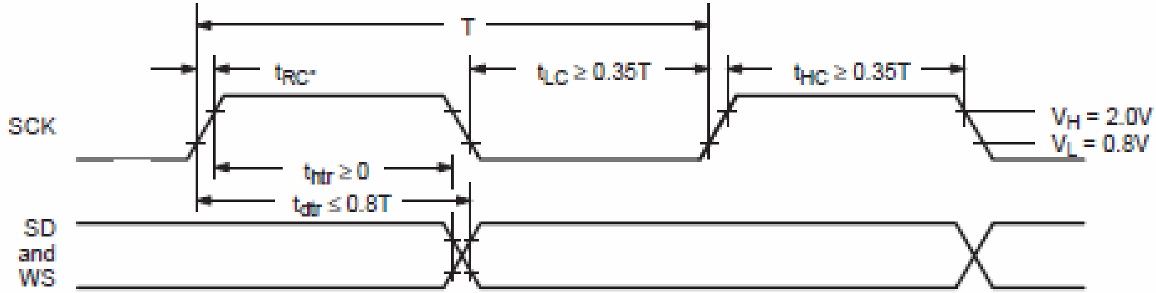
I²S interface

- Features, cont'd:
  - On the I2S bus there is only one transmitter and one monitoring device (the master)
  - The master can be a transmitter, a receiver or a system of supervising transmission between two slave devices
  - I2S interface transferres data of two channels: left and right
  - Data channels are transmitted on change
  - A large number of devices used for the transmission of additional supervisory controller

I²S configurations



I²S - time dependencies (transmitter)

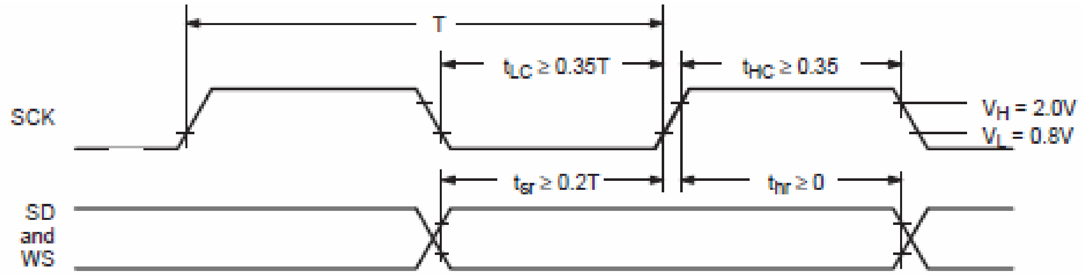


T = clock period
T_{tr} = minimum allowed clock period for transmitter
T > T_{tr}
* t_{RC} is only relevant for transmitters in slave mode.

Example: Master transmitter with data rate of 2.5MHz (±10%) (all values in ns)

	MIN	TYP	MAX	CONDITION
clock period T	360	400	440	T_{tr} = 360
clock HIGH t_{HC}	160			min > 0.35T = 140 (at typical data rate)
clock LOW t_{LC}	160			min > 0.35T = 140 (at typical data rate)
delay t_{dtr}			300	max < 0.80T = 320 (at typical data rate)
hold time t_{htr}	100			min > 0
clock rise-time t_{RC}			60	max > 0.15T_{tr} = 54 (only relevant in slave mode)

I²S - time dependencies (receiver)

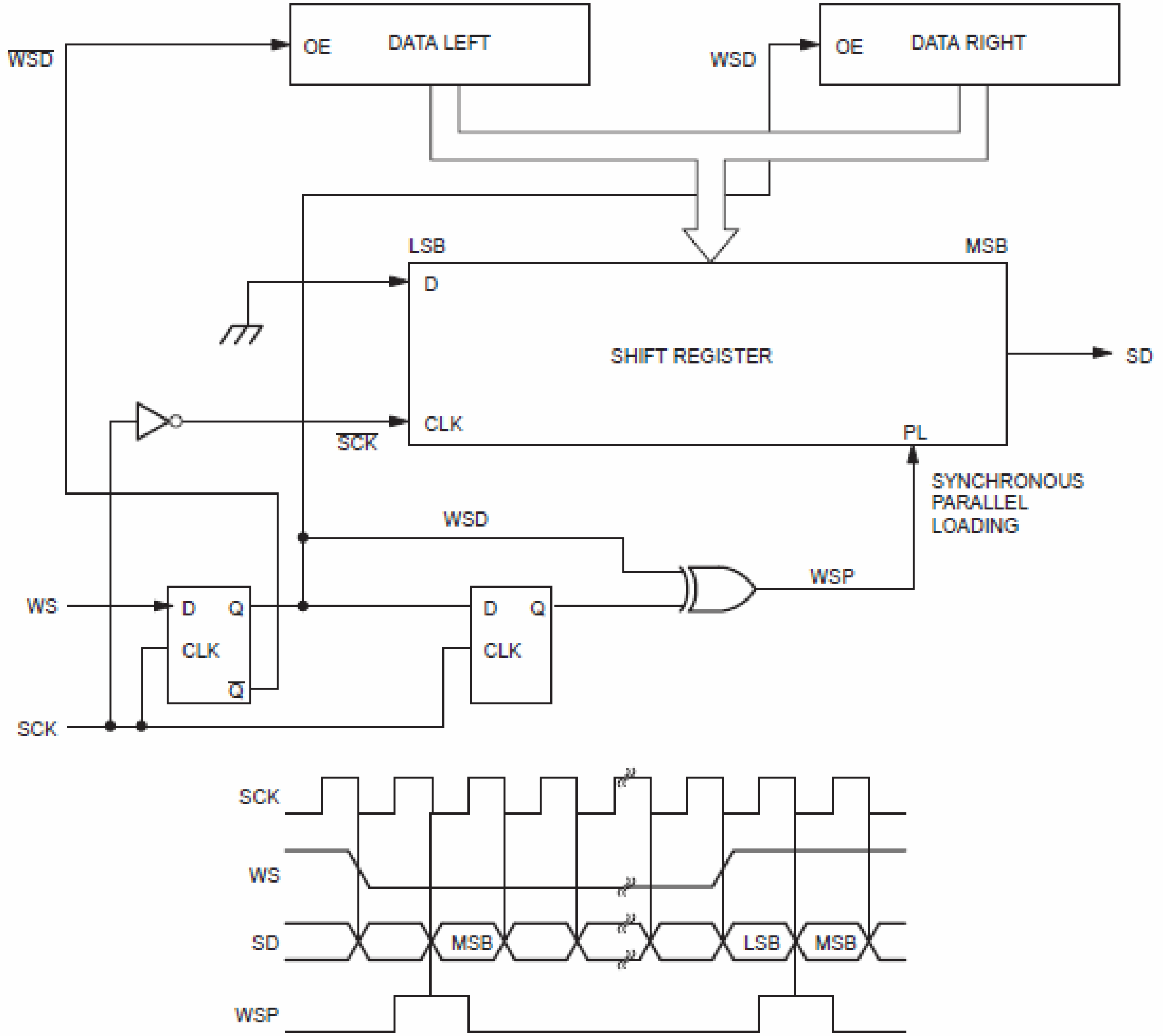


T = clock period
T_r = minimum allowed clock period for transmitter
T > T_r

Example: Slave receiver with data rate of 2.5MHz (±10%) (all values in ns)

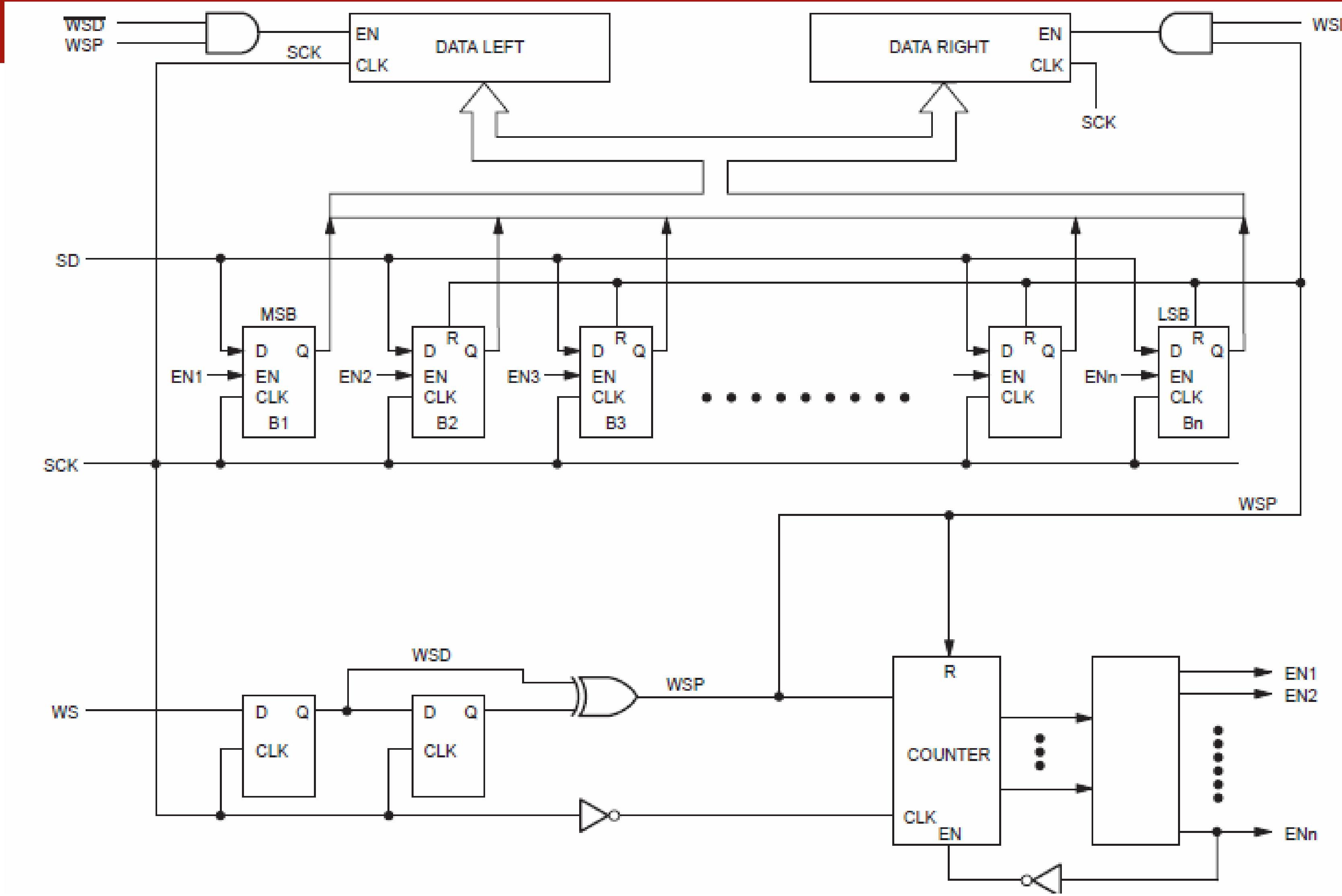
	MIN	TYP	MAX	CONDITION
clock period T	360	400	440	$T_{tr} = 360$
clock HIGH t_{HC}	110			min < 0.35T = 126
clock LOW t_{LC}	110			min < 0.35T = 126
set-up time t_{sr}	60			min < 0.20T = 72
hold time t_{htr}	0			min < 0

I²S - transmitter diagram

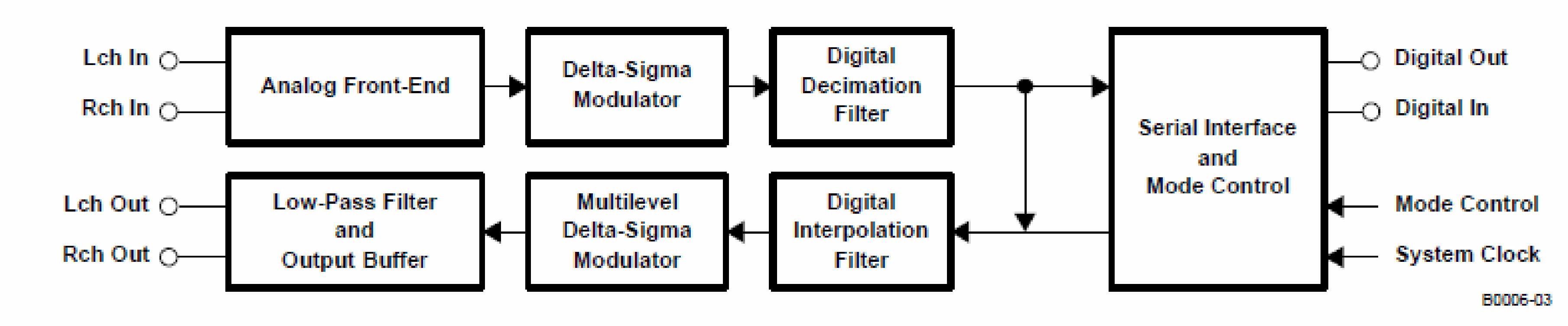


I²S - konstrukcja odbiornika

I²S - receiver diagram

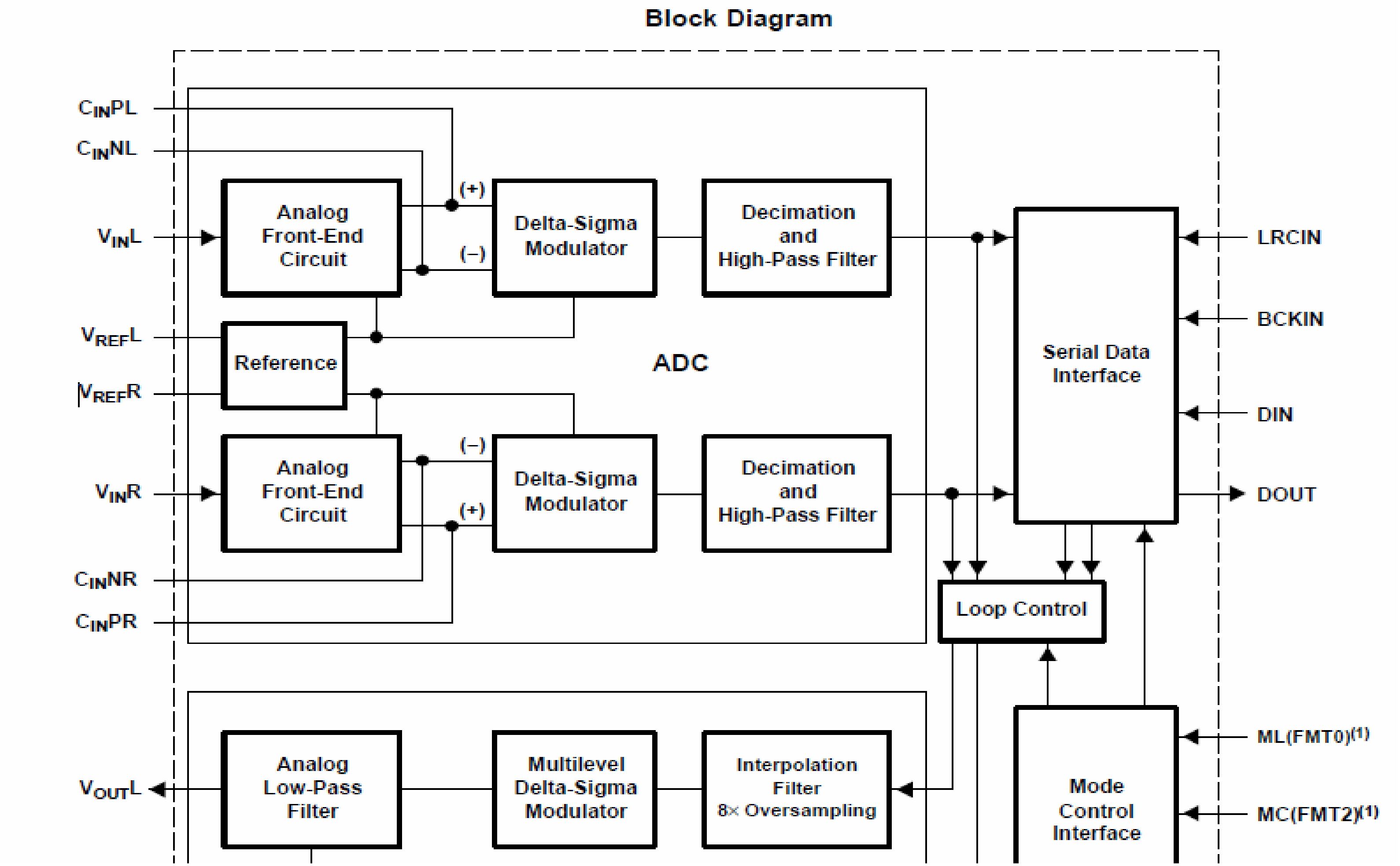


PCM3001 - Stereo Codec with analog output

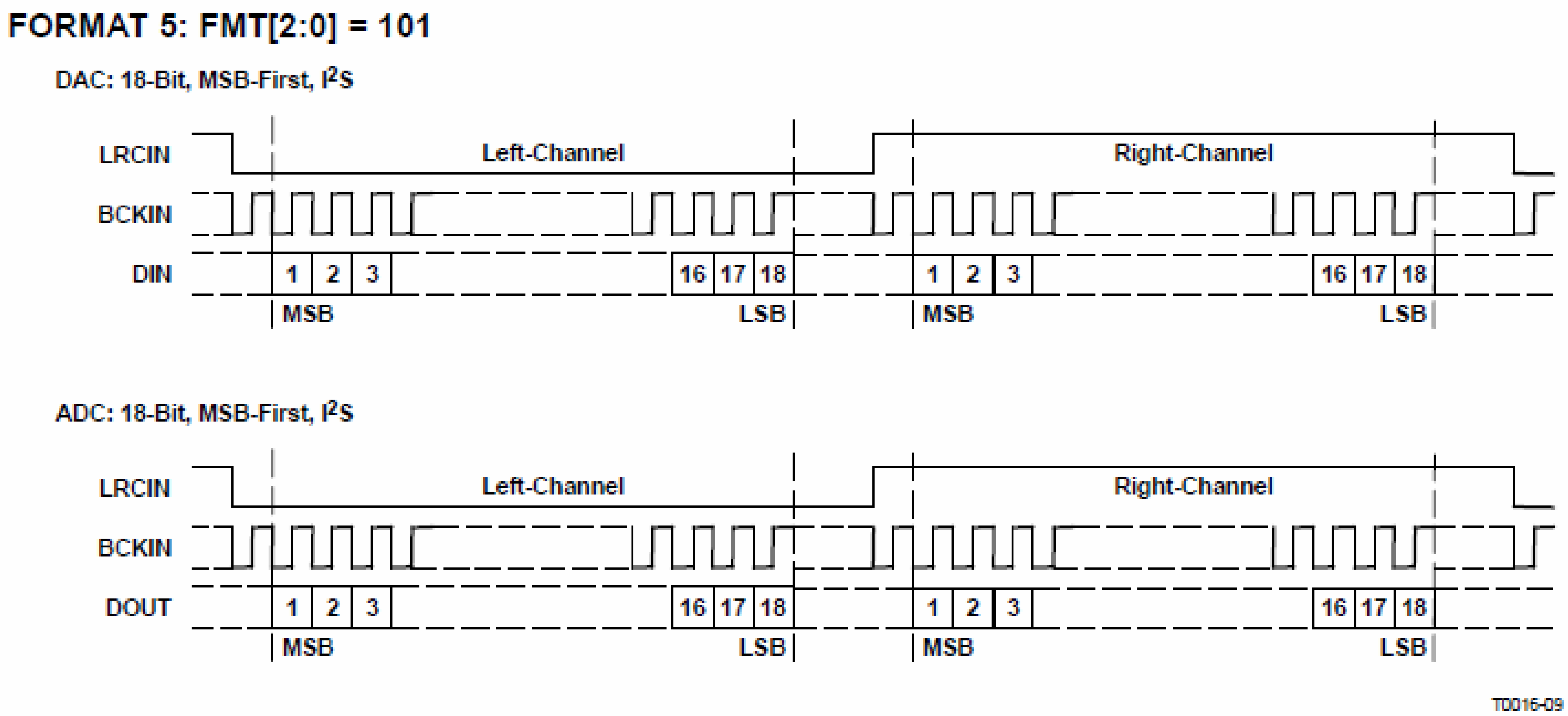


- *Low-cost system encoder / decoder audio*
- *Includes digital filters, provides digital signal attenuation, De-emphasis, mild blanking signal (soft-mute) and the detection of silence*
- *Control via a digital interface*

PCM3001 – Block diagram



PCM3001 - I²S


FORMAT 5: FMT[2:0] = 101
DAC: 18-Bit, MSB-First, I2S
LRCIN
Left-Channel
Right-Channel
BCKIN
DIN
1 2 3
16 17 18
MSB
LSB
ADC: 18-Bit, MSB-First, I2S
LRCIN
BCKIN
DOUT
T0016-09

Thank you for your attention

References

[1] Reference Manual RM0090, www.st.com