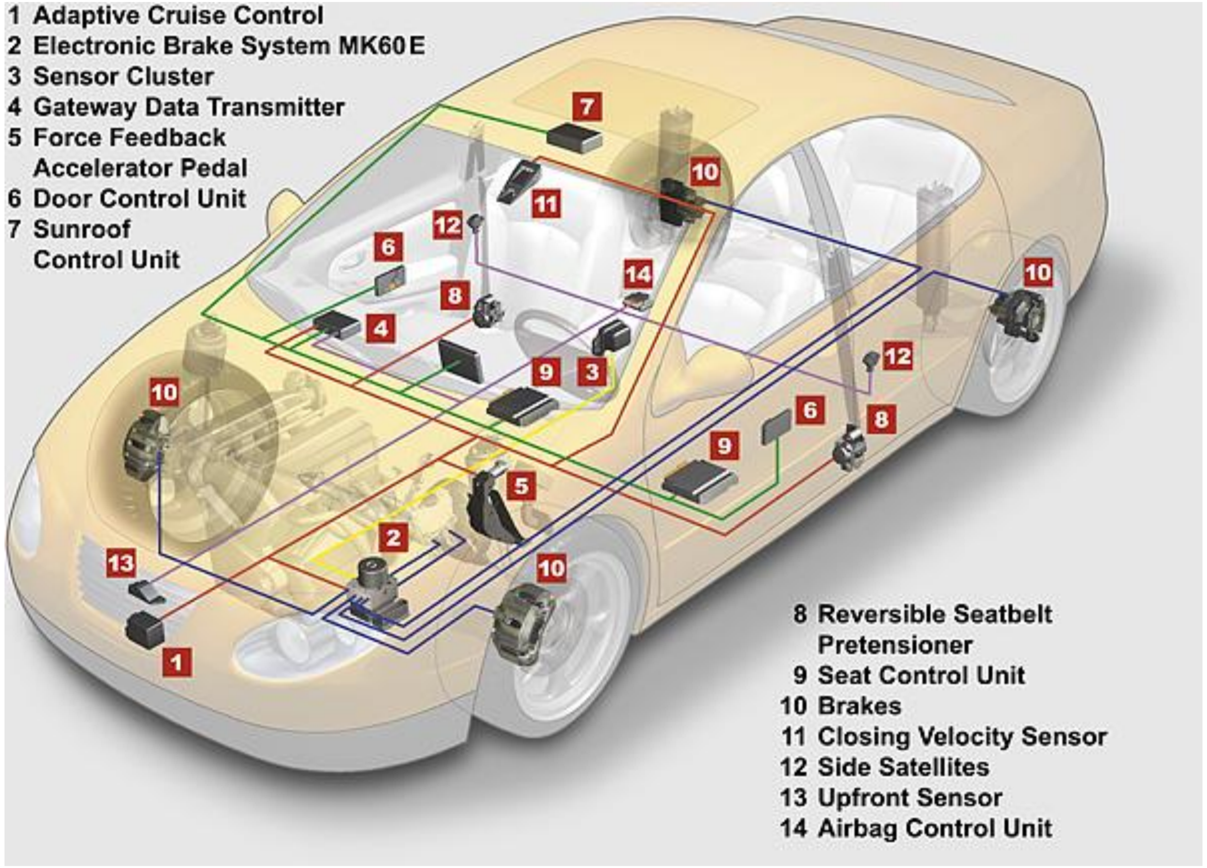

1 Adaptive Cruise Control
2 Electronic Brake System MK60E
3 Sensor Cluster
4 Gateway Data Transmitter
5 Force Feedback Accelerator Pedal
6 Door Control Unit
7 Sunroof Control Unit
8 Reversible Seatbelt Pretensioner
9 Seat Control Unit
10 Brakes
11 Closing Velocity Sensor
12 Side Satellites
13 Upfront Sensor
14 Airbag Control Unit
1
2
3
4
5
6
6
7
8
8
9
9
10
10
10
10
11
12
12
13
14

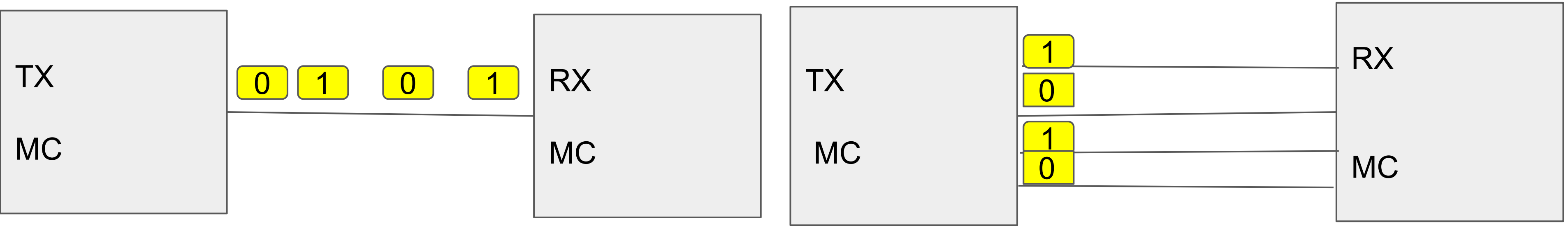
Communication

Medium:

	Wired	Wireless
SPEED	~	~
Safety	↑	↓
Cost	↓	↑
Installation	↓	↑
Mobility	↓	↑

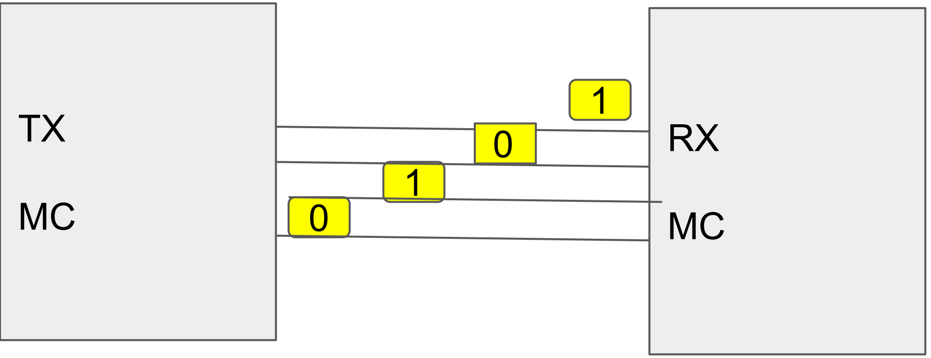
Transmission

1

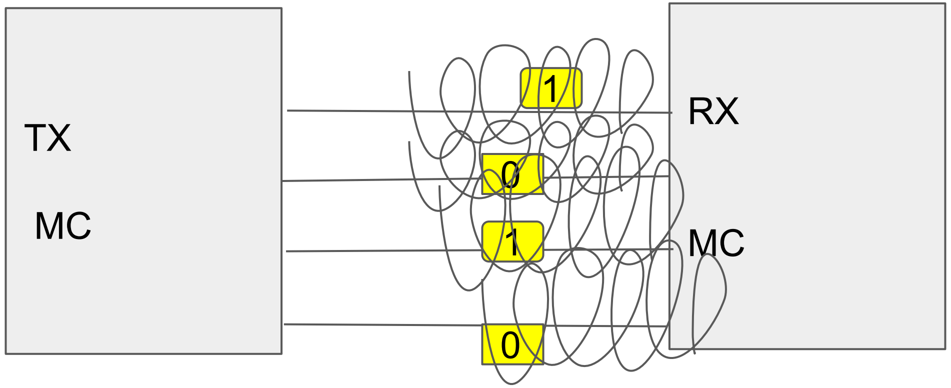


Parallel

Data Skew:

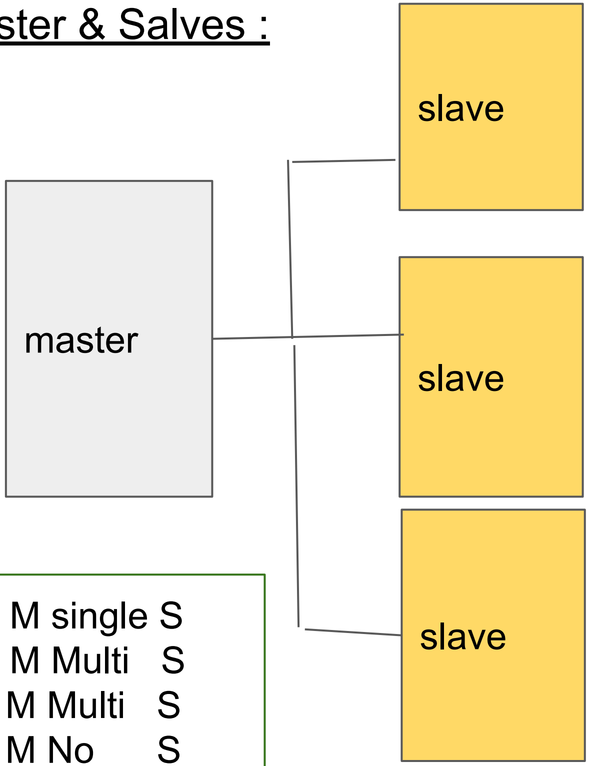


Back EmF:



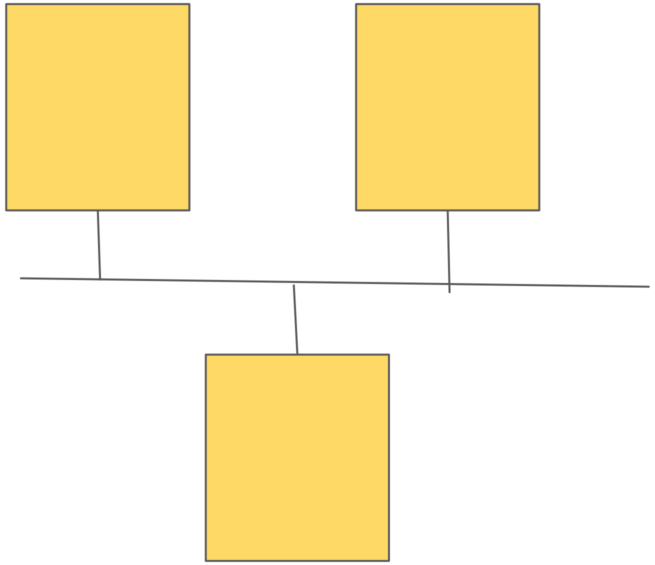
TX , RX Relations:

Master & Salves :



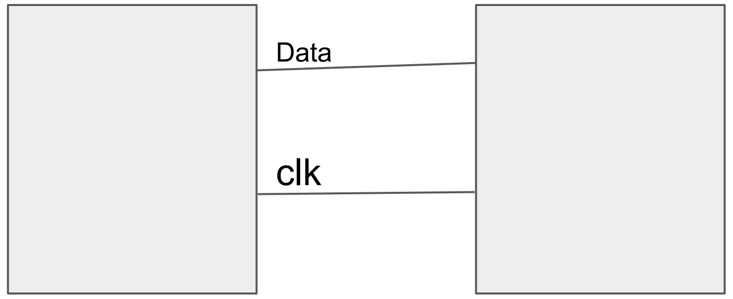
Single M single S
Single M Multi S
Multi M Multi S
Multi M No S

Peer to Peer :

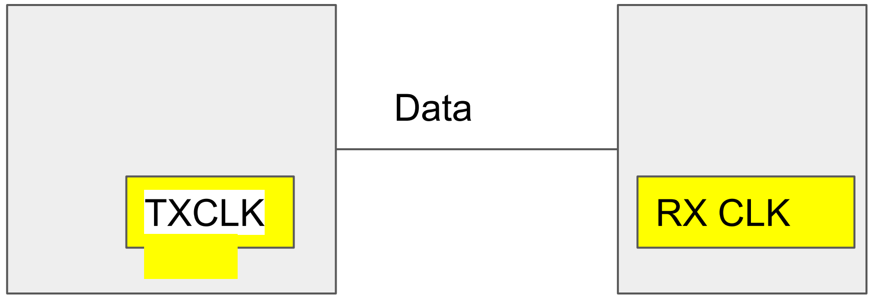


Syncronization :

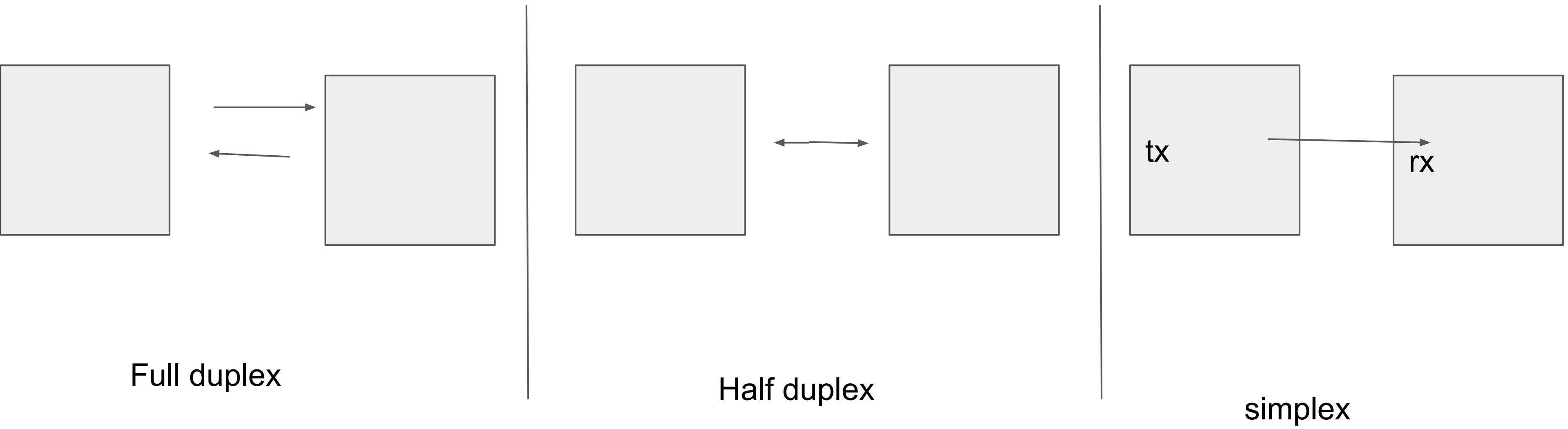
Syncronous :



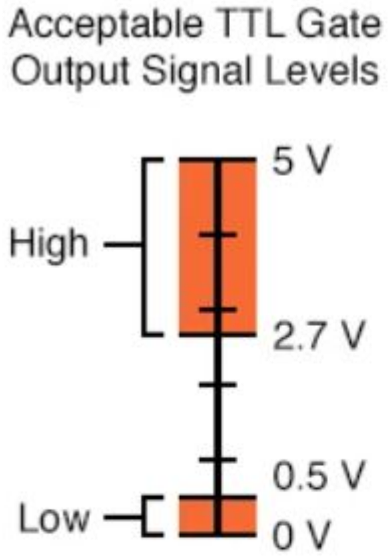
ASyncronous

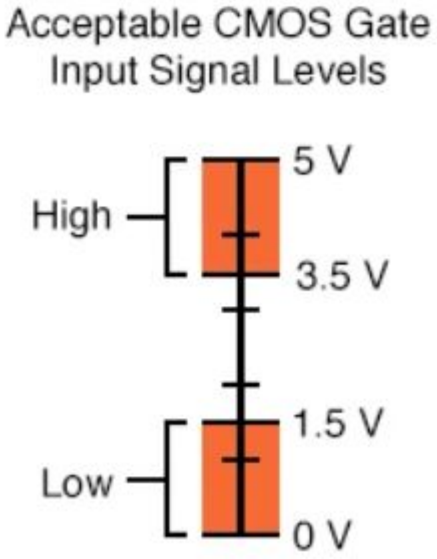


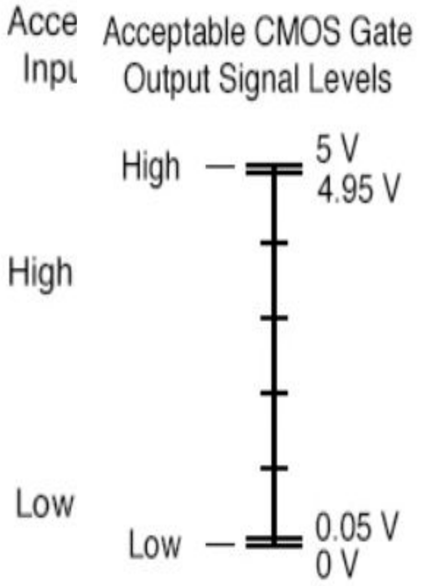
Data Direction:

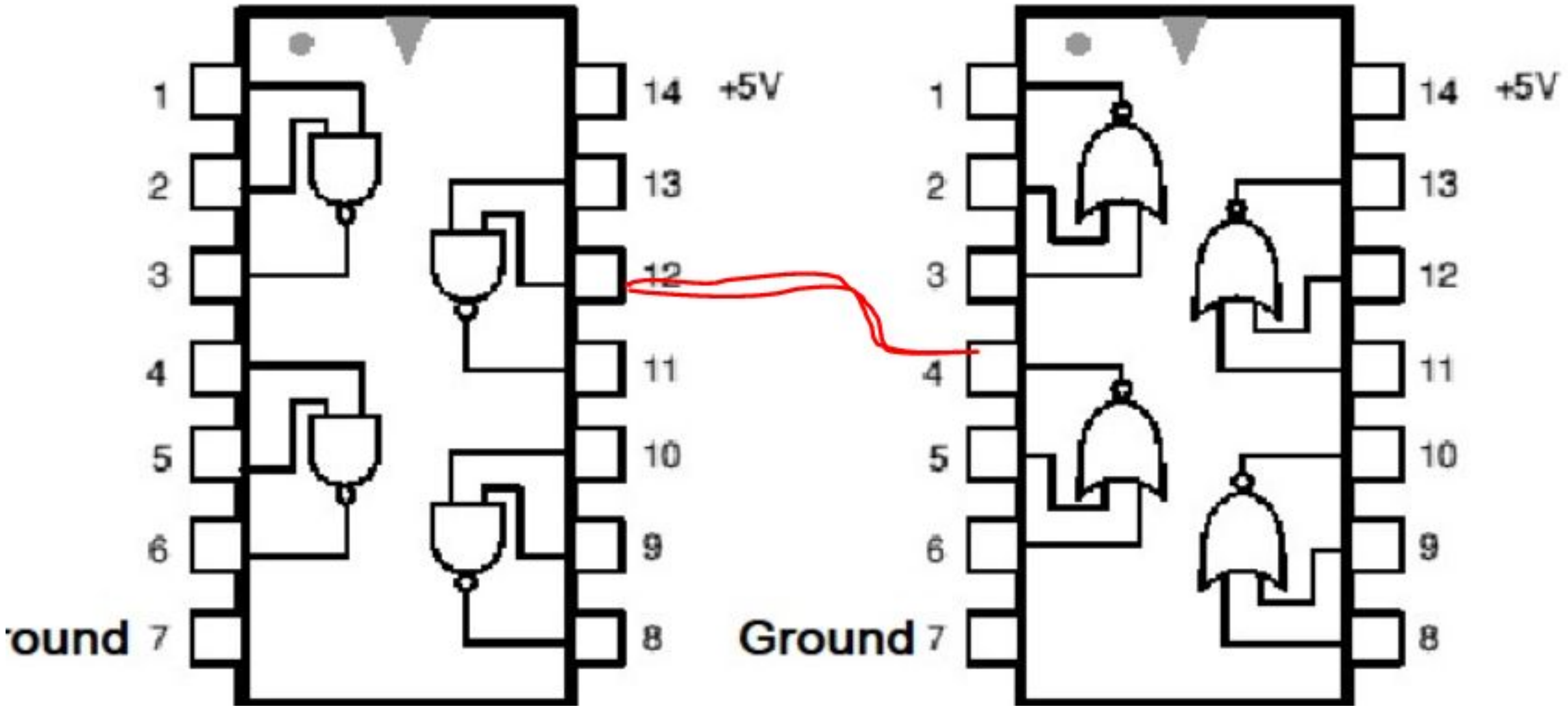


TTL VS CMOS:


Acceptable TTL Gate
Output Signal Levels
High
Low
5 V
2.7 V
0.5 V
0 V


Acceptable CMOS Gate
Input Signal Levels
High
Low
5 V
3.5 V
1.5 V
0 V


Acce
Inpu
Acceptable CMOS Gate
Output Signal Levels
High
High
Low
Low
5 V
4.95 V
0.05 V
0 V


1
2
3
4
5
6
round 7
14 +5V
13
12
11
10
9
8
1
2
3
4
5
6
Ground 7
14 +5V
13
12
11
10
9
8

Throughput:

Frame = (Data Bits + Another Bits (start+Stop+parity))

TP=DATA Bits /Frames Bits .

TP= 100% Frame is only Data.

TP= 0% Frame has no Data.

Throughput:

Frame = (Data Bits + Another Bits (start+Stop+parity))

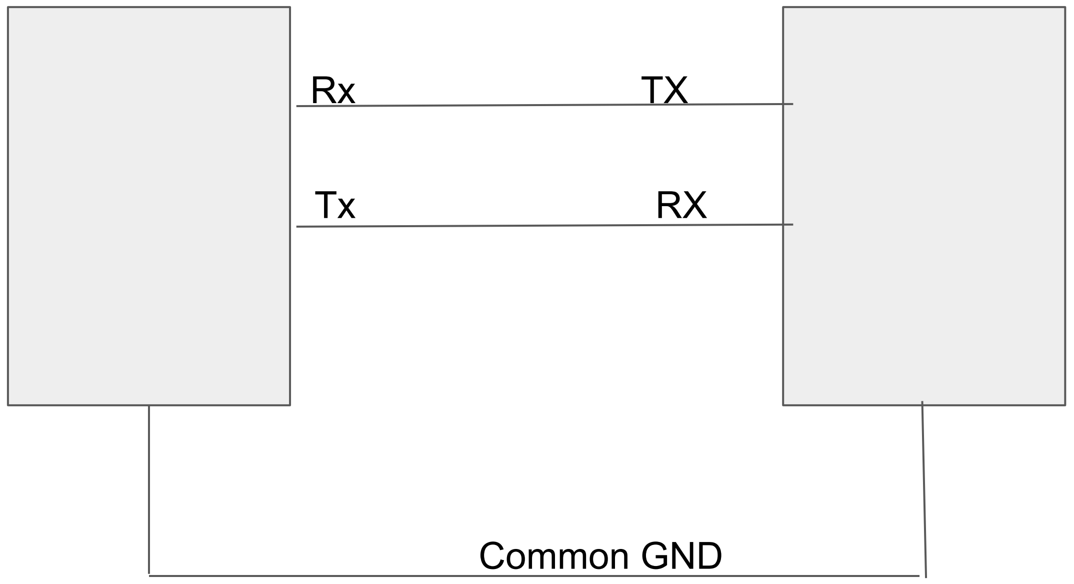
TP=DATA Bits /Frames Bits .

TP= 100% Frame is only Data.

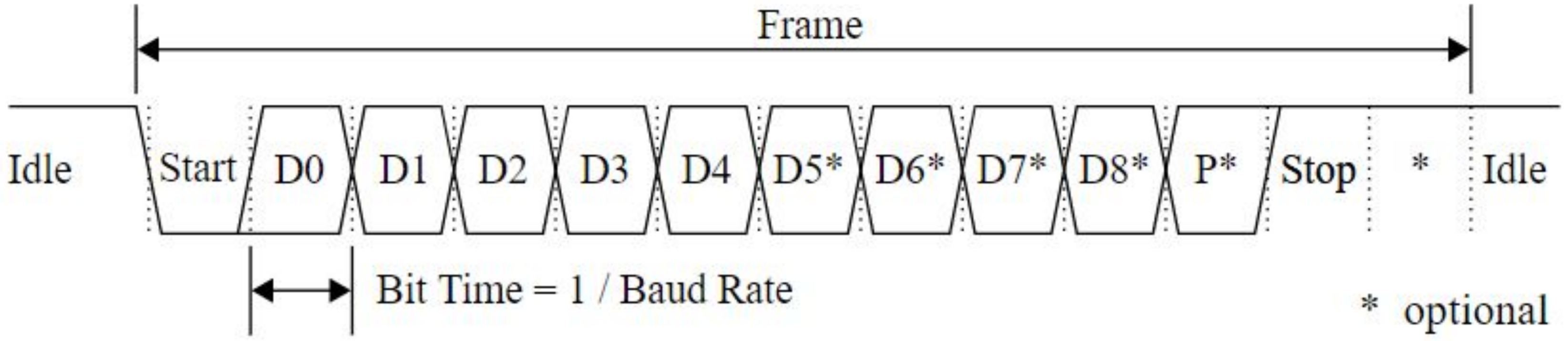
TP= 0% Frame has no Data.

UART:

- ❑ Wired
- ❑ Serial
- ❑ Peer to peer
- ❑ Async
- ❑ H full duplex
- ❑ Support 5,9,7,8,6
- ❑ Data overrun detection
- ❑ Frame error detection
- ❑ Double speed Async Mode



Data Frame format:



Parity by hardware =
Even or ODD

- ❏ Can’t detect error location
- ❏ Can’t detect multiple error

- ❏ 1,2 Stop bit

UART Meuasurment:

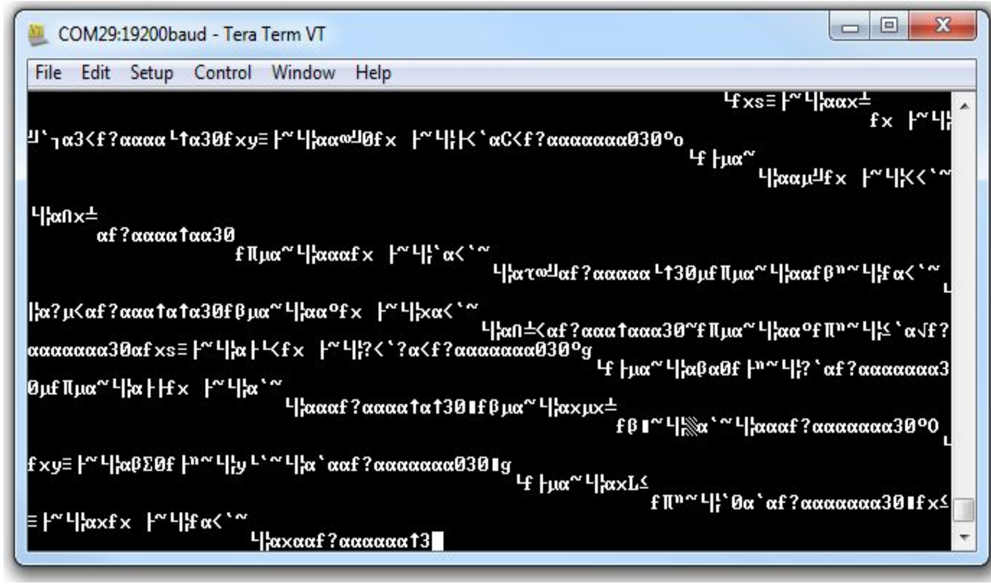
Max Throughput = (1 start+9 data bits+1 stop)=9/11=81%

Min Throughput =(1 start+ 5 data bits+1 parity+ 2 stop bits)= 5/9= 55%

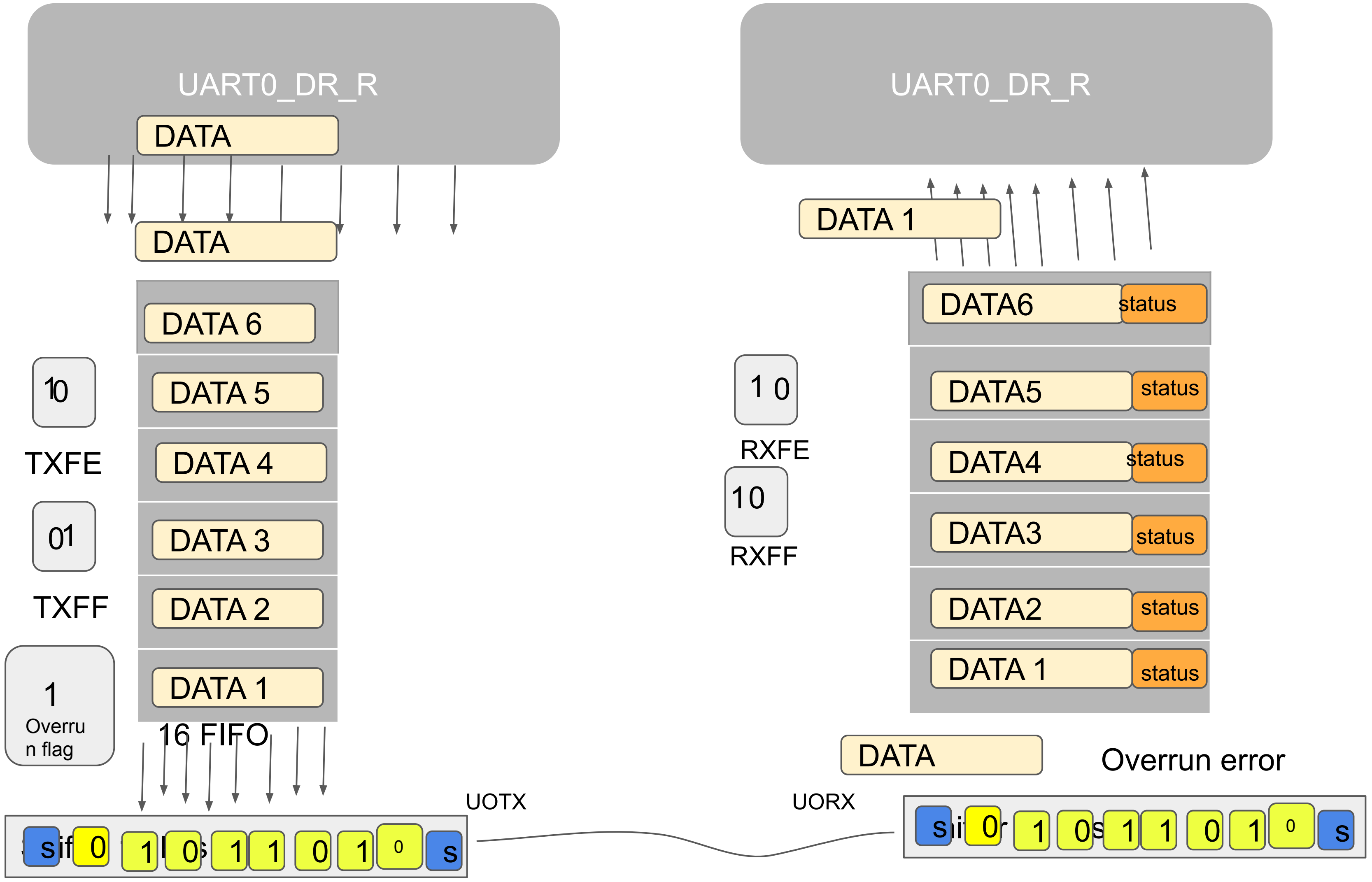
Baud rate = bite rate = number of bits per second

Baud rate :

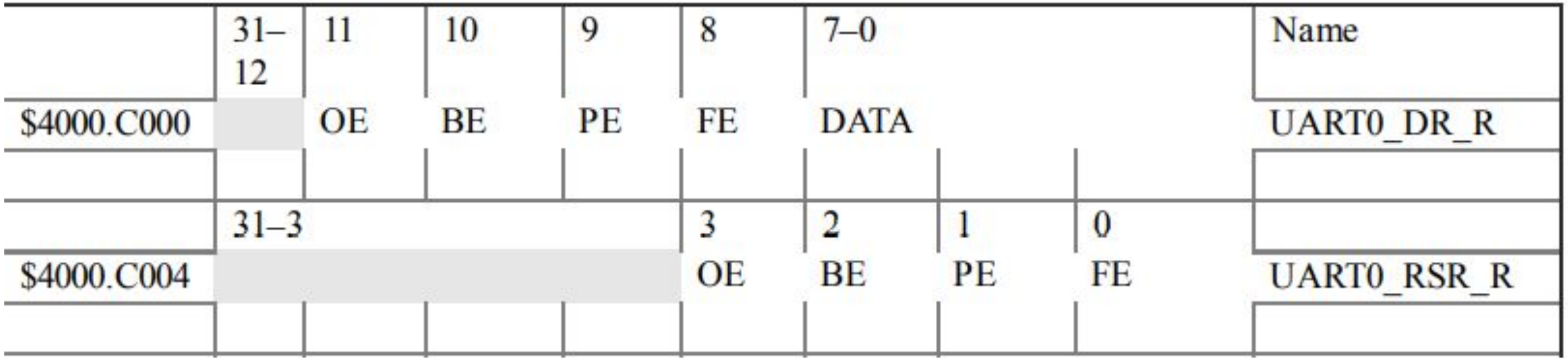
9600bps, 57,600 bps , 4800 bps, 115,200 bps


COM29:19200baud - Tera Term VT
File Edit Setup Control Window Help

```
                                                                      ╘fxs≡├~╡╞ααx±
                                                                              fx ├~╡╞
╝`┐α3<f?αααα└↑α30fxy≡├~╡╞αα╜0fx ├~╡╞├<`αC<f?ααααααα030°o
                                                                    ╘f├µα~
                                                                       ╡╞ααµ╜fx ├~╡╞<`~
╡╞αՈx±
   αf?αααα↑αα30
          fПµα~╡╞αααfx ├~╡╞`α<`~
                                         ╡╞ατ╜αf?ααααα└↑30µfПµα~╡╞ααfβ"~╡╞fα<`~
╞α?µ<αf?ααα↑α↑α30fβµα~╡╞αα°fx ├~╡╞xα<`~
                                         ╡╞αՈ±<αf?ααα↑ααα30~fПµα~╡╞αα°fП"~╡╞`α√f?
ααααααα30αfxs≡├~╡╞α├└<fx ├~╡╞?<`?α<f?ααααααα030°g
                                                    ╘f├µα~╡╞αβα0f├"~╡╞?`αf?ααααααα3
0µfПµα~╡╞α├├fx ├~╡╞α`~
                   ╡╞αααf?αααα↑α↑30▮fβµα~╡╞αxµx±
                                                  fβ▮~╡╞▒α`~╡╞αααf?ααααααα30°0
fxy≡├~╡╞αβΣ0f├"~╡╞y└`~╡╞α`ααf?ααααααα030▮g
                                              ╘f├µα~╡╞αxL≤
                                                   fП"~╡╞`0α`αf?ααααααα30▮fx≤
≡├~╡╞αxfx ├~╡╞fα<`~
             ╡╞αxααf?αααααα↑3█
```

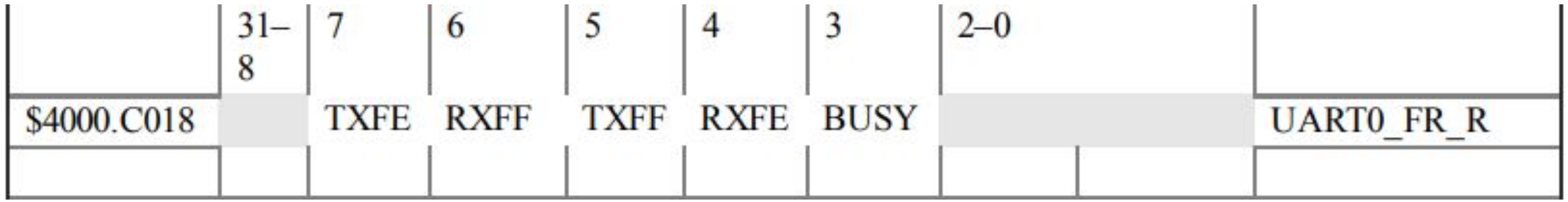

UART0_DR_R
DATA
DATA
DATA 6
DATA 5
DATA 4
DATA 3
DATA 2
DATA 1
16 FIFO
10
TXFE
01
TXFF
1
Overrun flag
UOTX
s
0
1
0
1
1
0
1
0
s
UART0_DR_R
DATA 1
DATA6 status
DATA5 status
DATA4 status
DATA3 status
DATA2 status
DATA 1 status
1 0
RXFE
10
RXFF
DATA
Overrun error
UORX
s
0
1
0
1
1
0
1
0
s

	31–12	11	10	9	8	7–0			Name
$4000.C000		OE	BE	PE	FE	DATA			UART0_DR_R
	31–3				3	2	1	0	
$4000.C004					OE	BE	PE	FE	UART0_RSR_R
	31–8	7	6	5	4	3	2–0		
$4000.C018		TXFE	RXFF	TXFF	RXFE	BUSY			UART0_FR_R
	31–16	15–0							
$4000.C024		DIVINT							UART0_IBRD_R
	31–6				5–0				
$4000.C028					DIVFRAC				UART0_FBRD_R
	31–8	7	6 – 5	4	3	2	1	0	
$4000.C02C		SPS	WPEN	FEN	STP2	EPS	PEN	BRK	UART0_LCRH_R
	31–10	9	8	7	6–3	2	1	0	
$4000.C030		RXE	TXE	LBE		SIRLP	SIREN	UARTEN	UART0_CTL_R



	31–12	11	10	9	8	7–0			Name
$4000.C000		OE	BE	PE	FE	DATA			UART0_DR_R
	31–3				3	2	1	0	
$4000.C004					OE	BE	PE	FE	UART0_RSR_R

- The OE, BE, PE, and FE are error flags associated with the receiver. You can see these flags in two places: associated with each data byte in UART0_DR_R or as a separate error register in UART0_RSR_R



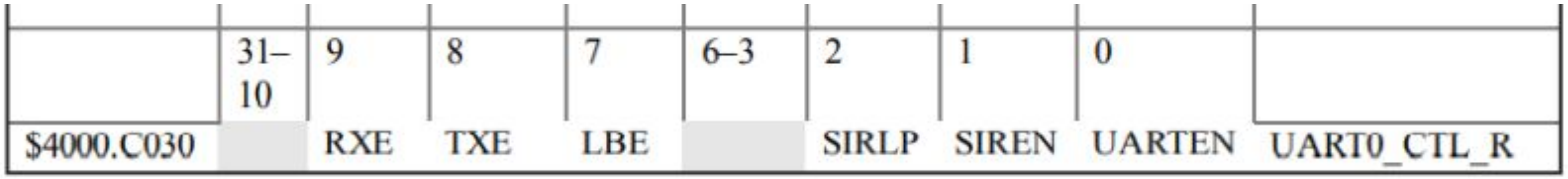
	31–8	7	6	5	4	3	2–0	
$4000.C018		TXFE	RXFF	TXFF	RXFE	BUSY		UART0_FR_R

- ❏ Busy : The BUSY flag is set when the transmitter still has unsent bits.
- ❏ Busy : The BUSY flag is cleared when last stop bit has been sent and transmit FIFO is empty

- ❏ RXFE: The RXFE flag is set when receive shift register and receive FIFO are initially empty
- ❏ RXFE: The RXFE flag is cleared when first frame goes into the receive FIFO.

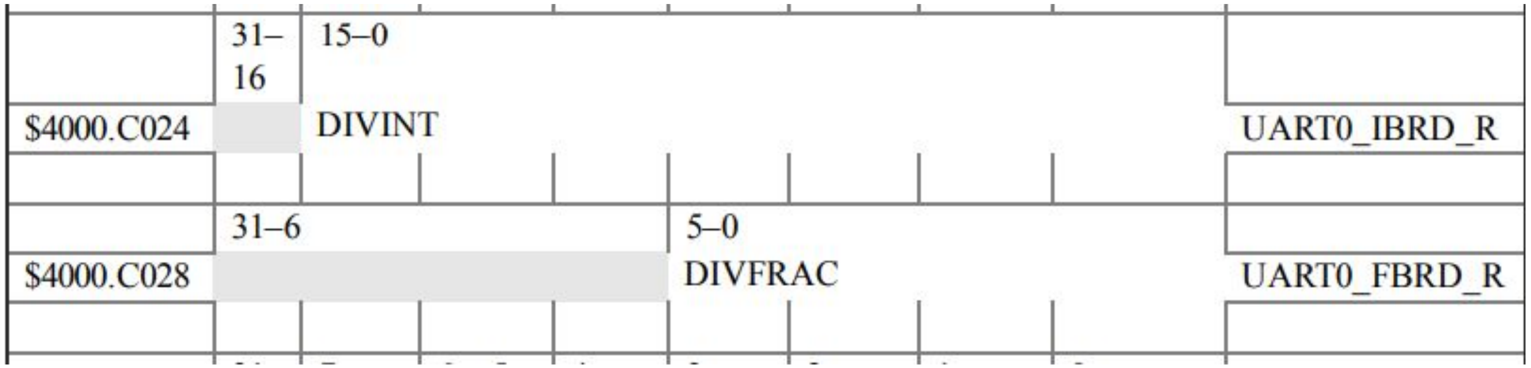
- ❏ RXFF: The RXFE flag is set when when the FIFO is full
- ❏ RXFF: The RXFE flag is cleared when FIFO is not full

- ❏ TXFF: The RXFE flag is set when when the FIFO is full
- ❏ TXFF: The RXFE flag is cleared when FIFO is not full



	31–10	9	8	7	6–3	2	1	0	
$4000.C030		RXE	TXE	LBE		SIRLP	SIREN	UARTEN	UART0_CTL_R

- The UART0_CTL_R control register contains the bits that turn on the UART. TXE is the Transmitter Enable bit, and RXE is the Receiver Enable bit. We set TXE, RXE, and UARTEN equal to 1 in order to activate the UART device. However, we should clear UARTEN during the initialization sequence.



	31–16	15–0	
$4000.C024		DIVINT	UART0_IBRD_R
	31–6	5–0	
$4000.C028		DIVFRAC	UART0_FBRD_R

- ❑ The UART0_IBRD_R and UART0_FBRD_R registers specify the baud rate.
- ❑ The Baud16 clock is created from the system bus clock, with a frequency of (Bus clock frequency)/divider
- ❑ The baud rate is 16 times slower than Baud16
- ❑ Baud rate = Baud16/16 = (Bus clock frequency)/(16*divider)

Example:

d= 8/16/1920

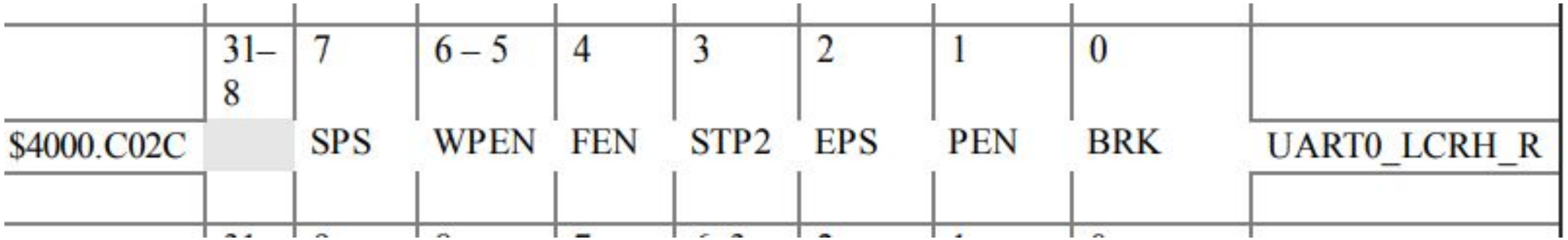
-bus clock is 8 MHz   - 19200 bits /sec desired Baudrate.

19200= 8x10^6 /16*d

d= 26.04167= 11010.000011 , d= 26 , (0.04167 *64+0.5)=3.1668 ---> 3

UART0_IBRD_R= 11010.

UART0_FBRD_R=000011.



	31–8	7	6 – 5	4	3	2	1	0	
$4000.C02C		SPS	WPEN	FEN	STP2	EPS	PEN	BRK	UART0_LCRH_R

- UART0_LCRH_R , UART0_IBRD_R , and UART0_FBRD_R form an internal 30-bit register. This internal register is only updated when a write operation to UART0_LCRH_R is performed
- 8-bit word length, enable FIFO

Sheet 6

Q1. Assume System clock frequency=16MHz. Find the values for the divisor registers of UARTIBRD and UARTFBRD for the following standard baud rates:

(a) 4800 (b) 9600 (c) 57,600 (d) 115,200

By default, 16 MHz System Clock is divided by 16 before it is fed to the UART.

Therefore, Divisor= 16MHz/(16*BaudRate) = 1MHz/BaudRate.

(a) 1MHz/4800 = 208.3333, UARTIBRD = 208 and UARTFBRD = (0.3333×64) + 0.5 = 21.8312 = 21

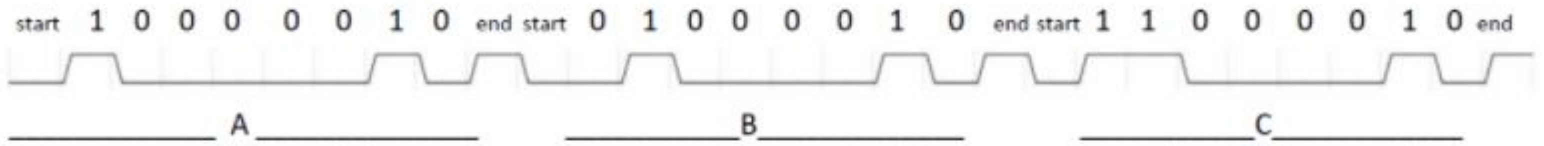
(b) 1MHz/9600 = 104.166666, UARTIBRD = 104 and UARTFBRD = (0.16666 × 64) +0.5=11

(c) 1MHz/57600 = 17.361, UARTIBRD = 17 and UARTFBRD = (0.361 × 64) + 0.5 =23

(d) 1MHz/115200 = 8.680, UARTIBRD = 8 and UARTFBRD = (0.680 × 64) +0.5=44

Q2. Assume the baud rate is 9600 bits/sec. Show the serial port output versus time waveform that occurs when the ASCII characters "ABC" are transmitted one right after another. What is the total time to transmit the three characters?

A 65 -> 0100 0001
B 66 -> 0100 0010
C 67 -> 0100 0011


start 1 0 0 0 0 0 1 0 end start 0 1 0 0 0 0 1 0 end start 1 1 0 0 0 0 1 0 end
A
B
C

Each char has 8 bits+ start bit + end bit = 10 bits.
Time = (10*3) * bit-time = 30 / baud rate = 30 / 9600 = 3.125 mS.

Q3. Write a C function to initialize UART0 with baud rate 9600 bits/s, 8 bits word length, no parity, one stop bit, and FIFO enabled.

```
 void UART_Init(void){ // should be called only once

SYSCTL_RCGCUART_R |= 0x0001; // activate UART0

SYSCTL_RCGCGPIO_R |= 0x0001; // activate port A

UART0_CTL_R &= ~0x0001; // disable UART

UART0_IBRD_R = 520; // IBRD=int(80000000/(16*9600)) = int(520.8333)

UART0_FBRD_R = 53; // FBRD = int(0.8333 * 64 + 0.5)

UART0_LCRH_R = 0x0070; // 8-bit word length, enable FIFO 001110000

 UART0_CTL_R = 0x0301; // enable RXE, TXE and UART 001100000001

GPIO_PORTA_AFSEL_R |= 0x03; // enable alt function PA0 ,PA1
GPIO_PORTA_PCTL_R = (GPIO_PORTA_PCTL_R&0xFFFFFF00)+0x00000011; /*
GPIO_PORTA_DEN_R |= 0x03; // enable digital I/O on PA0, PA1
GPIO_PORTA_AMSEL_R &= ~0x03; // disable analog function on PA0, PA1
```

Q4. Write a C function to check if there is data available to be received by UART0.

- ❏ **We need to check the empty flag of the receiver, if FIFO buffer is not empty, then there is data available to be received.**

```
bool UART0_Available(void){
return (UART0_FR_R & 0x010 != 0) ? 0 : 1;
}
```

Write a C program that receives from Device1 a lower-case character and transmits its upper-case toDevice2.

```
Void read_writeToUpper(void){
char in;
char out;
while(1){
in = UART0_Read();
out = in - 0x20; // To upper case (ex. a = A + 32)
UART0_Write(out);
}}
```

Q6. Write a C function to transmit one byte using UART0.

Q5. Write a C function to receive one byte using UART0.

```
void UART0_Write(char data){

while((UART0_FR_R & 0x0020) != 0); //check if the buffer is full

UART0_DR_R = data;

}
```