CAN Network

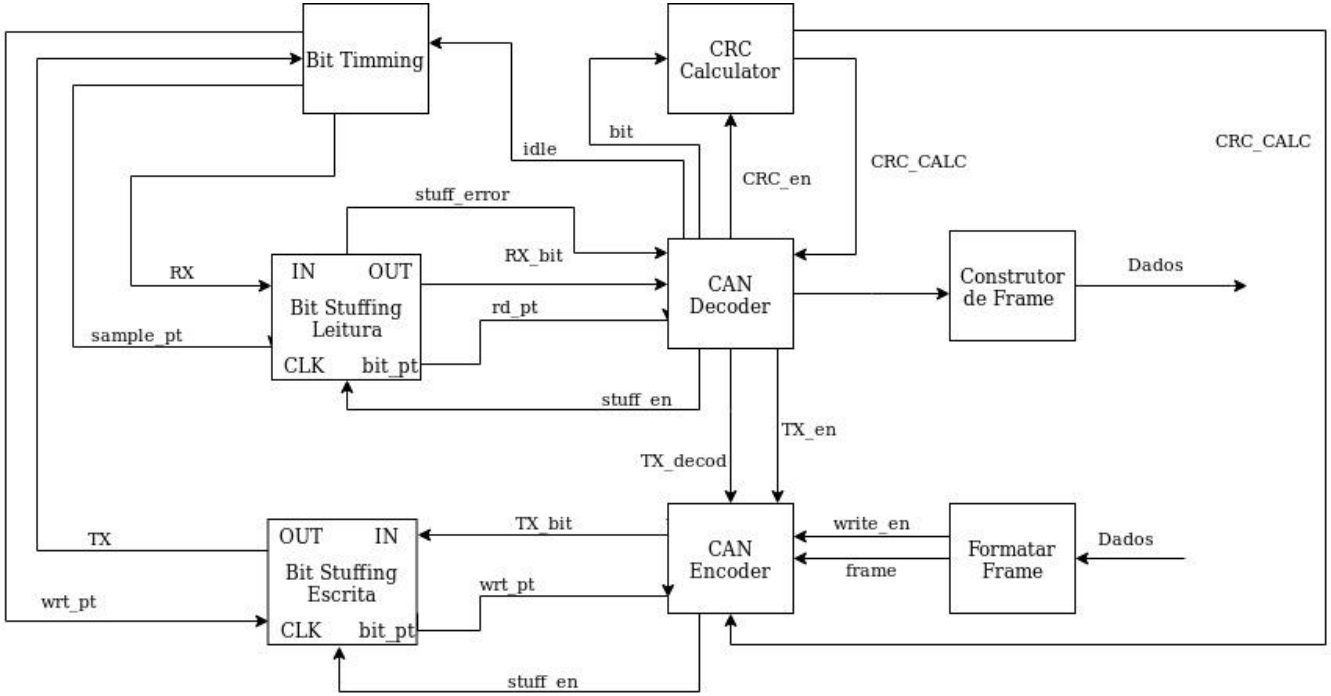
Project of Automotive Network Course

Students:
- Felipe Martins
- Lucas Cavalcanti
- Roberto Fernandes

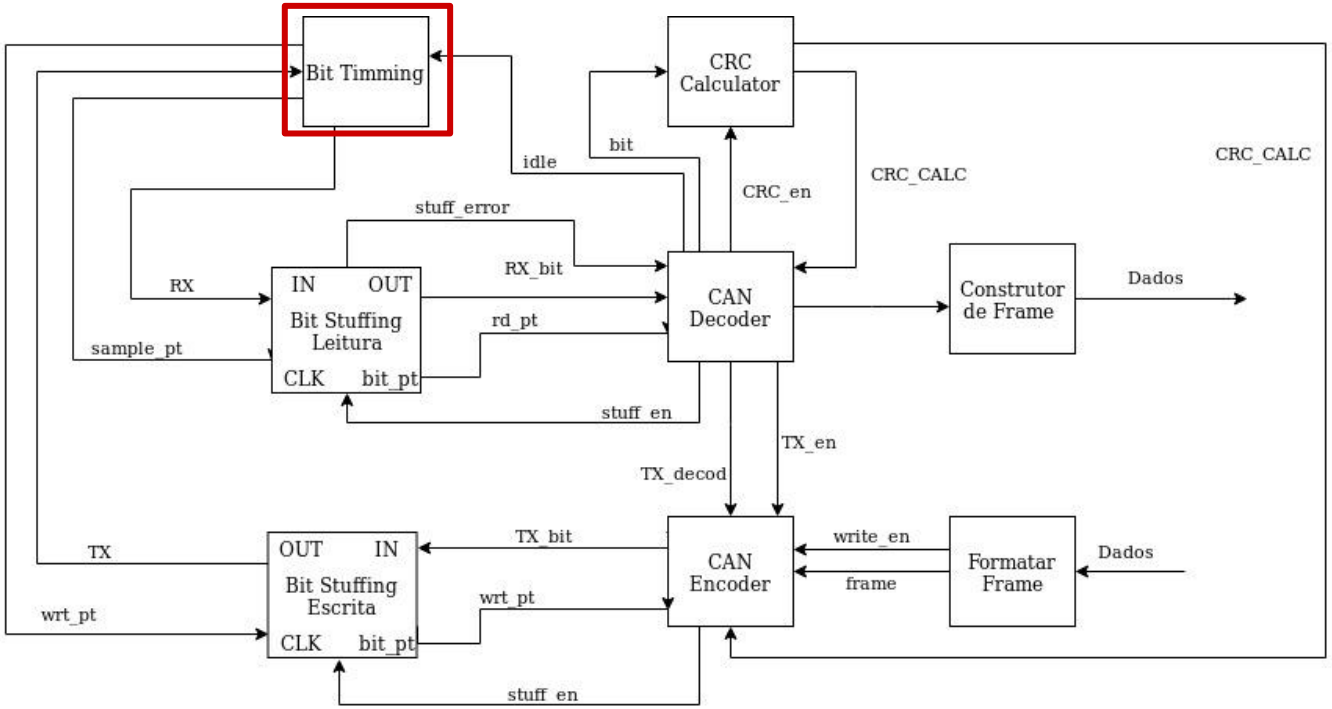
Summary

1. CAN Network
2. Bit Timing
3. Bit Stuffing
4. Decoder
5. Encoder
6. CRC
7. Formatar Frame
8. Testes Encoder
9. Testes Decoder

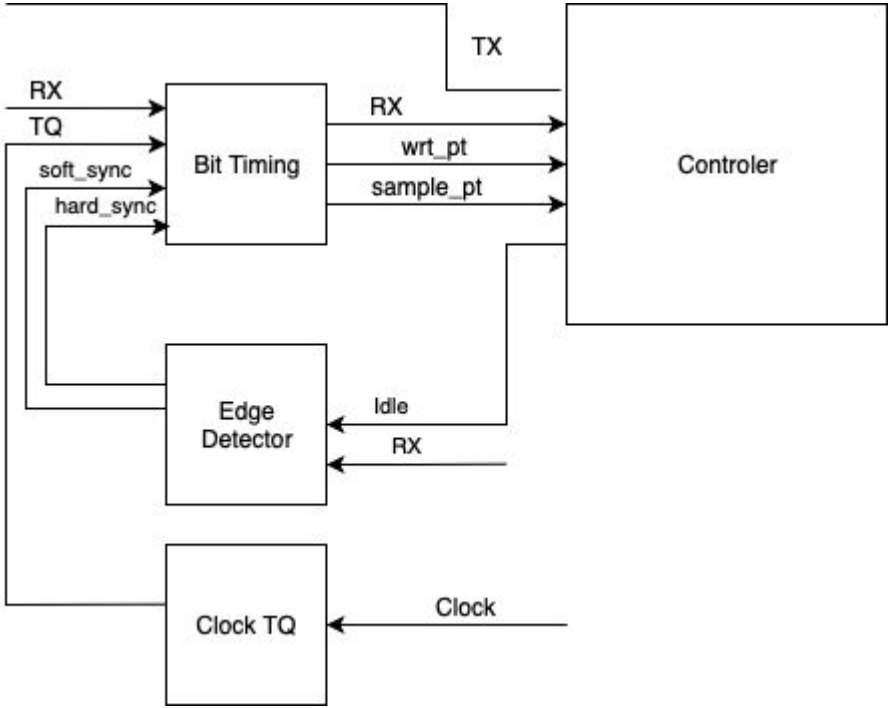
Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_en
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

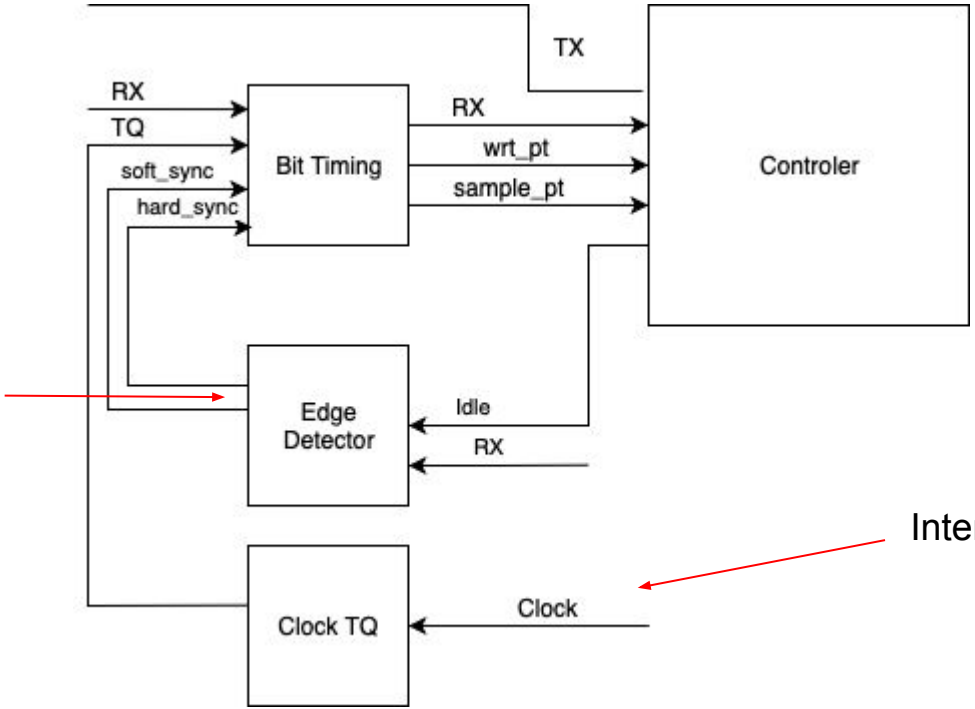
Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_en
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

Bit Timing


TX
RX
TQ
soft_sync
hard_sync
Bit Timing
RX
wrt_pt
sample_pt
Controler
Edge
Detector
Idle
RX
Clock TQ
Clock

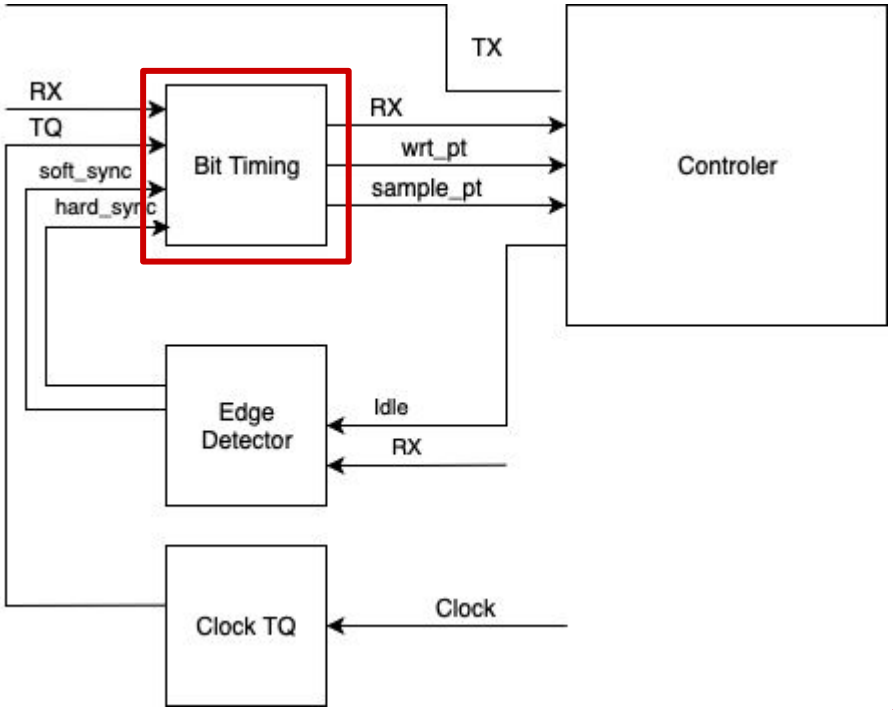
Bit Timing



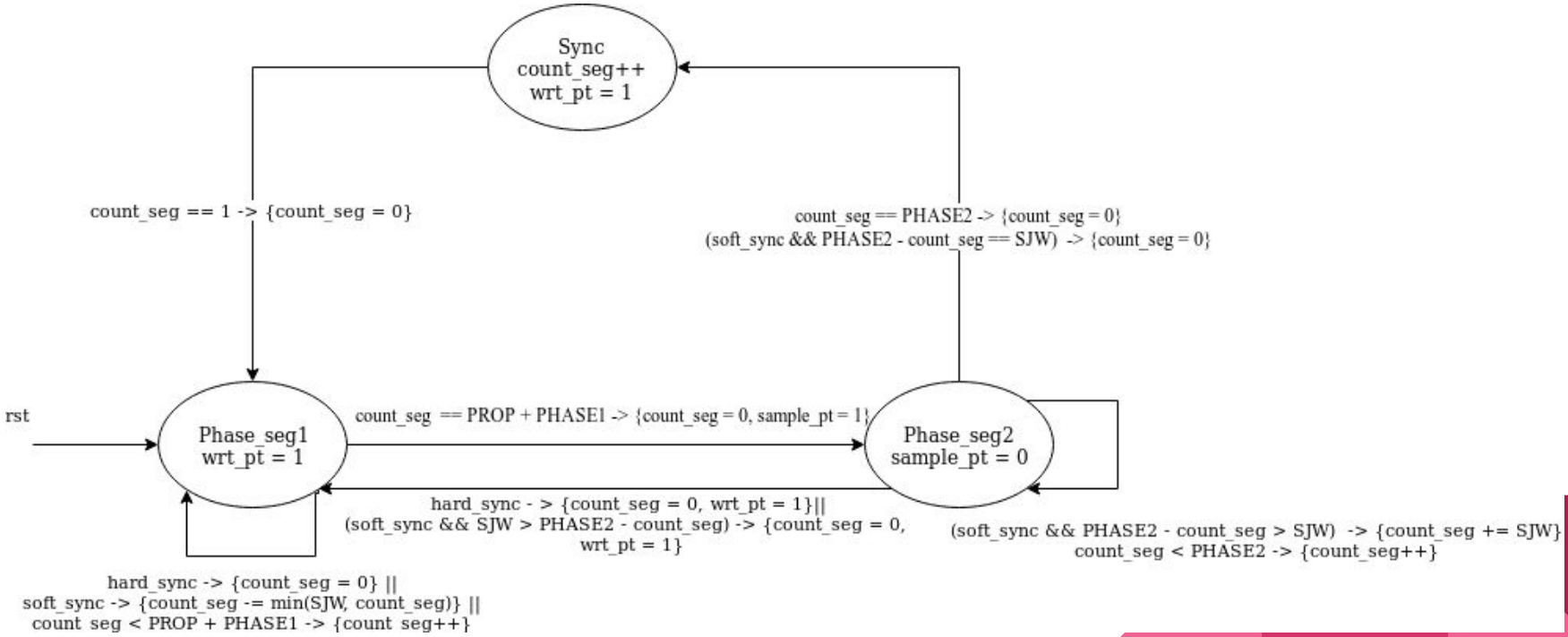
Interrupção Pino

Interrupção de Timer

Bit Timing


TX
RX
TQ
soft_sync
hard_sync
Bit Timing
RX
wrt_pt
sample_pt
Controler
Edge
Detector
Idle
RX
Clock TQ
Clock

Bit Timing

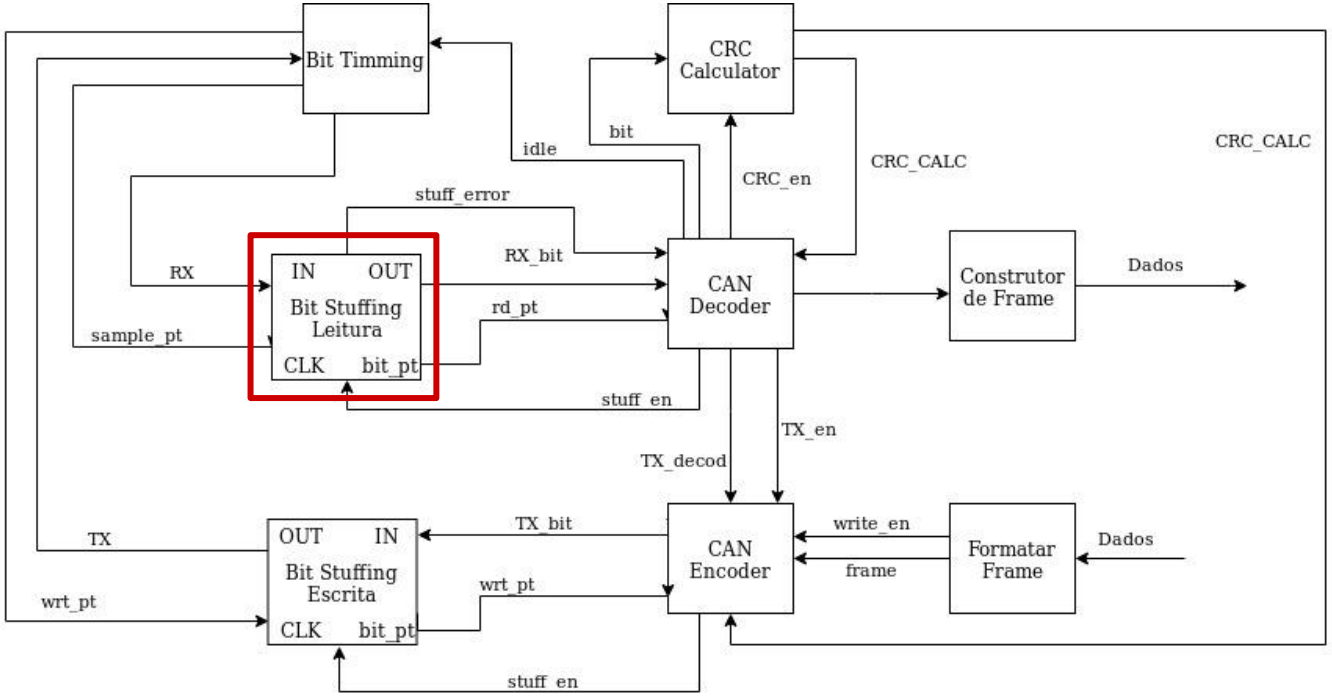

Sync
count_seg++
wrt_pt = 1
count_seg == 1 -> {count_seg = 0}
count_seg == PHASE2 -> {count_seg = 0}
(soft_sync && PHASE2 - count_seg == SJW) -> {count_seg = 0}
rst
Phase_seg1
wrt_pt = 1
count_seg == PROP + PHASE1 -> {count_seg = 0, sample_pt = 1}
Phase_seg2
sample_pt = 0
hard_sync - > {count_seg = 0, wrt_pt = 1}||
(soft_sync && SJW > PHASE2 - count_seg) -> {count_seg = 0,
wrt_pt = 1}
(soft_sync && PHASE2 - count_seg > SJW) -> {count_seg += SJW}
count_seg < PHASE2 -> {count_seg++}
hard_sync -> {count_seg = 0} ||
soft_sync -> {count_seg -= min(SJW, count_seg)} ||
count_seg < PROP + PHASE1 -> {count_seg++}

Bit Timing

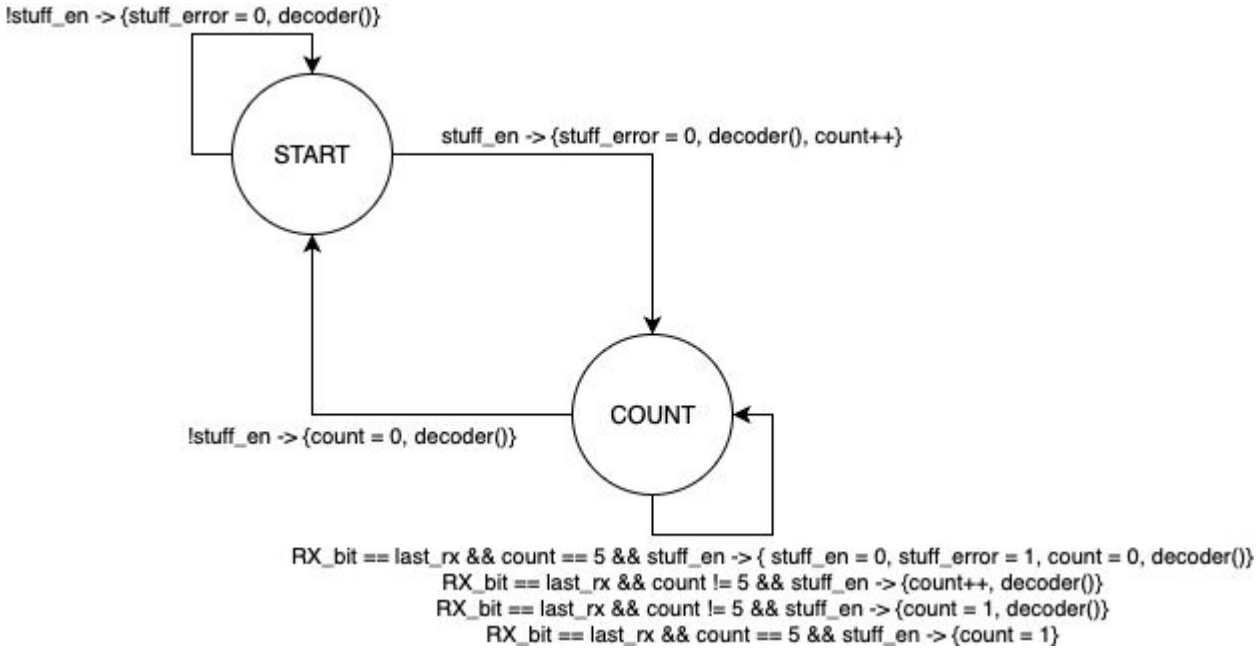
```
tq_clock.attach(bitTimingSM, TIME_QUANTA_S);
```

```
void bitTimingSM(){
  static int state = PHASE1_ST;
  static int count = 0;
  switch(state){
    case SYNC_ST:…
    case PHASE1_ST:…
    case PHASE2_ST:…
  }
```

Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_en
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

Bit Stuffing

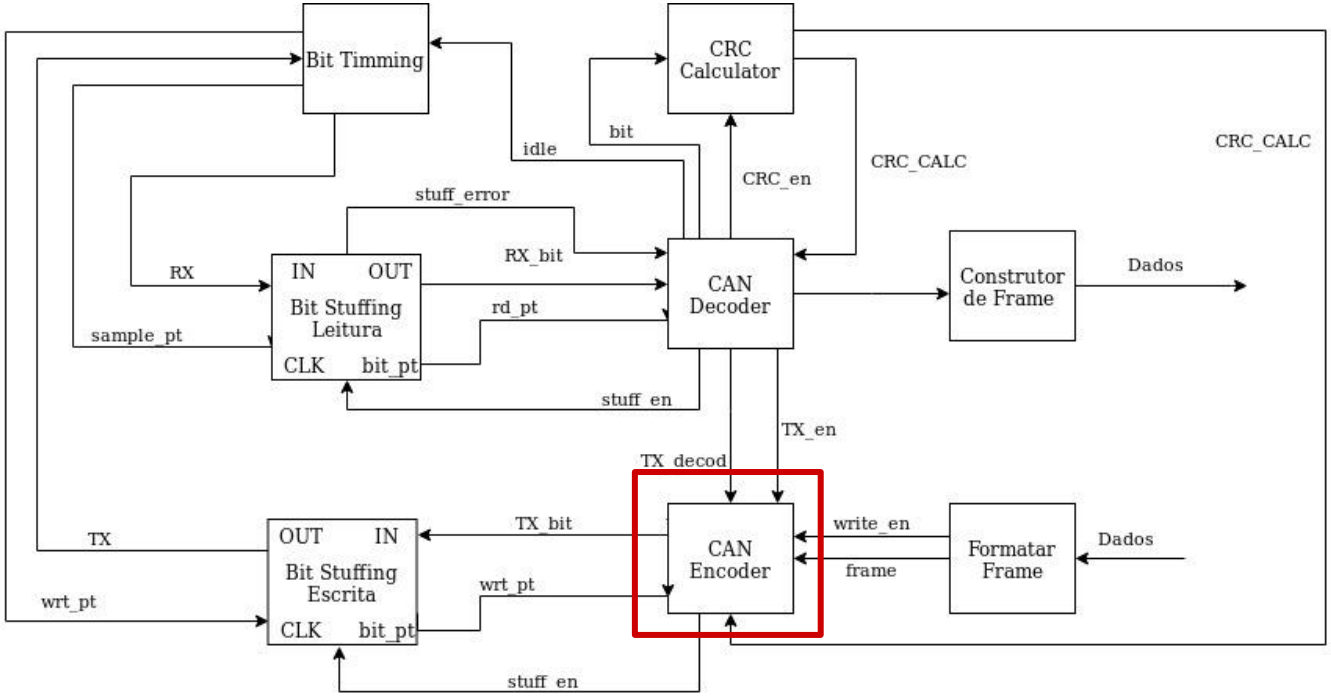


Bit Stuffing

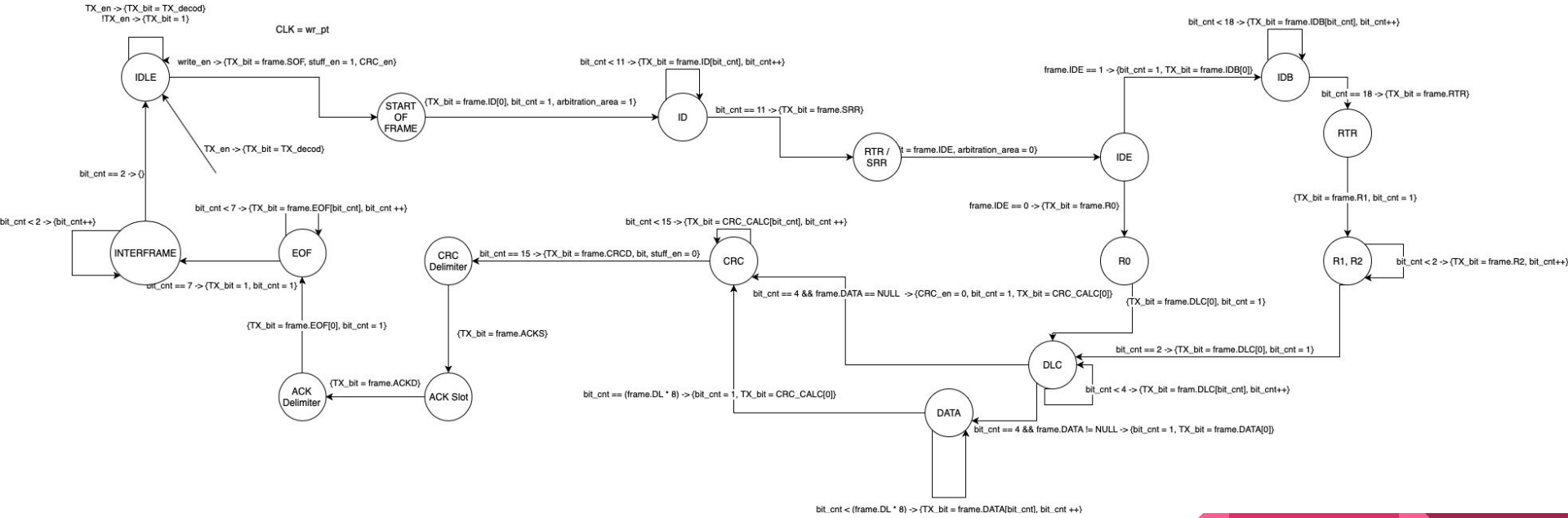
```
void bitstuffREAD()
{
  static int count = 0;
  static int state = 0;
  static int last_rx;
  last_rx = RX_bit;
  RX_bit = RX.read(); // TRANSCIEVER
  switch(state)
  {
    case(START):…
    case(COUNT):…
  }
}
```

```
case(START):…
case(COUNT):
  if(RX_bit == last_rx && count == 5 && stuff_en)
  {
    stuff_en = 0;
    stuff_error = 1;
    count = 0;
    decoder();
    state = COUNT;
  } else {
    if(!stuff_en){…
    }
    else if(RX_bit == last_rx) {…
    }
    else if(RX_bit != last_rx && count == 5) // STUFF
    {
      count = 1;
      debug(pc.printf("stuff - read\n"));
    }
    else if(RX_bit != last_rx && count != 5)
    {
      count = 1;
      decoder();
    }
  }
  break;
```

Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_en
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

Encoder


TX_en -> {TX_bit = TX_decod}
!TX_en -> {TX_bit = 1}
CLK = wr_pt
IDLE
write_en -> {TX_bit = frame.SOF, stuff_en = 1, CRC_en}
START OF FRAME
{TX_bit = frame.ID[0], bit_cnt = 1, arbitration_area = 1}
bit_cnt < 11 -> {TX_bit = frame.ID[bit_cnt], bit_cnt++}
ID
bit_cnt == 11 -> {TX_bit = frame.SRR}
RTR / SRR
= frame.IDE, arbitration_area = 0}
IDE
frame.IDE == 1 -> {bit_cnt = 1, TX_bit = frame.IDB[0]}
IDB
bit_cnt < 18 -> {TX_bit = frame.IDB[bit_cnt], bit_cnt++}
bit_cnt == 18 -> {TX_bit = frame.RTR}
RTR
{TX_bit = frame.R1, bit_cnt = 1}
R1, R2
bit_cnt < 2 -> {TX_bit = frame.R2, bit_cnt++}
frame.IDE == 0 -> {TX_bit = frame.R0}
R0
{TX_bit = frame.DLC[0], bit_cnt = 1}
DLC
bit_cnt == 2 -> {TX_bit = frame.DLC[0], bit_cnt = 1}
bit_cnt < 4 -> {TX_bit = fram.DLC[bit_cnt], bit_cnt++}
bit_cnt == 4 && frame.DATA != NULL -> {bit_cnt = 1, TX_bit = frame.DATA[0]}
DATA
bit_cnt < (frame.DL * 8) -> {TX_bit = frame.DATA[bit_cnt], bit_cnt ++}
bit_cnt == (frame.DL * 8) -> {bit_cnt = 1, TX_bit = CRC_CALC[0]}
bit_cnt == 4 && frame.DATA == NULL -> {CRC_en = 0, bit_cnt = 1, TX_bit = CRC_CALC[0]}
CRC
bit_cnt < 15 -> {TX_bit = CRC_CALC[bit_cnt], bit_cnt ++}
bit_cnt == 15 -> {TX_bit = frame.CRCD, bit, stuff_en = 0}
CRC Delimiter
{TX_bit = frame.ACKS}
ACK Slot
{TX_bit = frame.ACKD}
ACK Delimiter
{TX_bit = frame.EOF[0], bit_cnt = 1}
EOF
bit_cnt < 7 -> {TX_bit = frame.EOF[bit_cnt], bit_cnt ++}
bit_cnt == 7 -> {TX_bit = 1, bit_cnt = 1}
INTERFRAME
bit_cnt < 2 -> {bit_cnt++}
bit_cnt == 2 -> {}
TX_en -> {TX_bit = TX_decod}

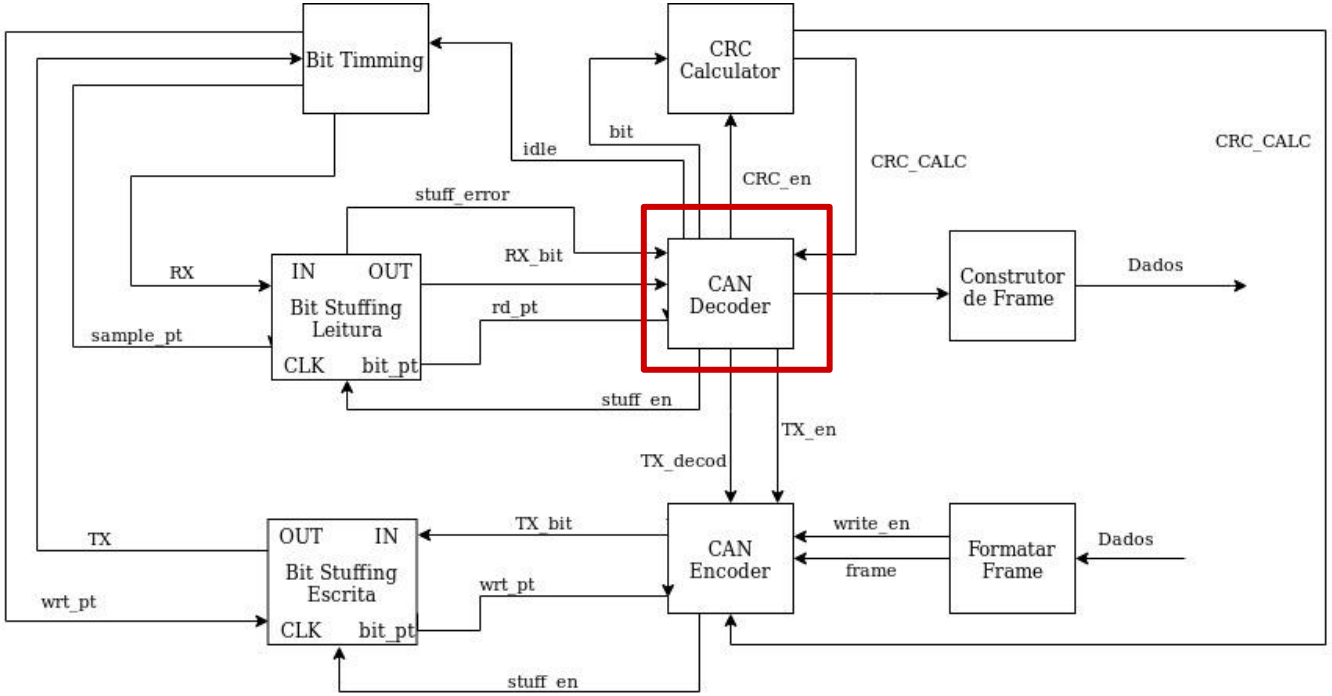
Encoder

```
void encoder(){
  static int state = 0;
  static int bit_cnt = 0;
  if(TX_en){
    state = IDLE;
  }
```

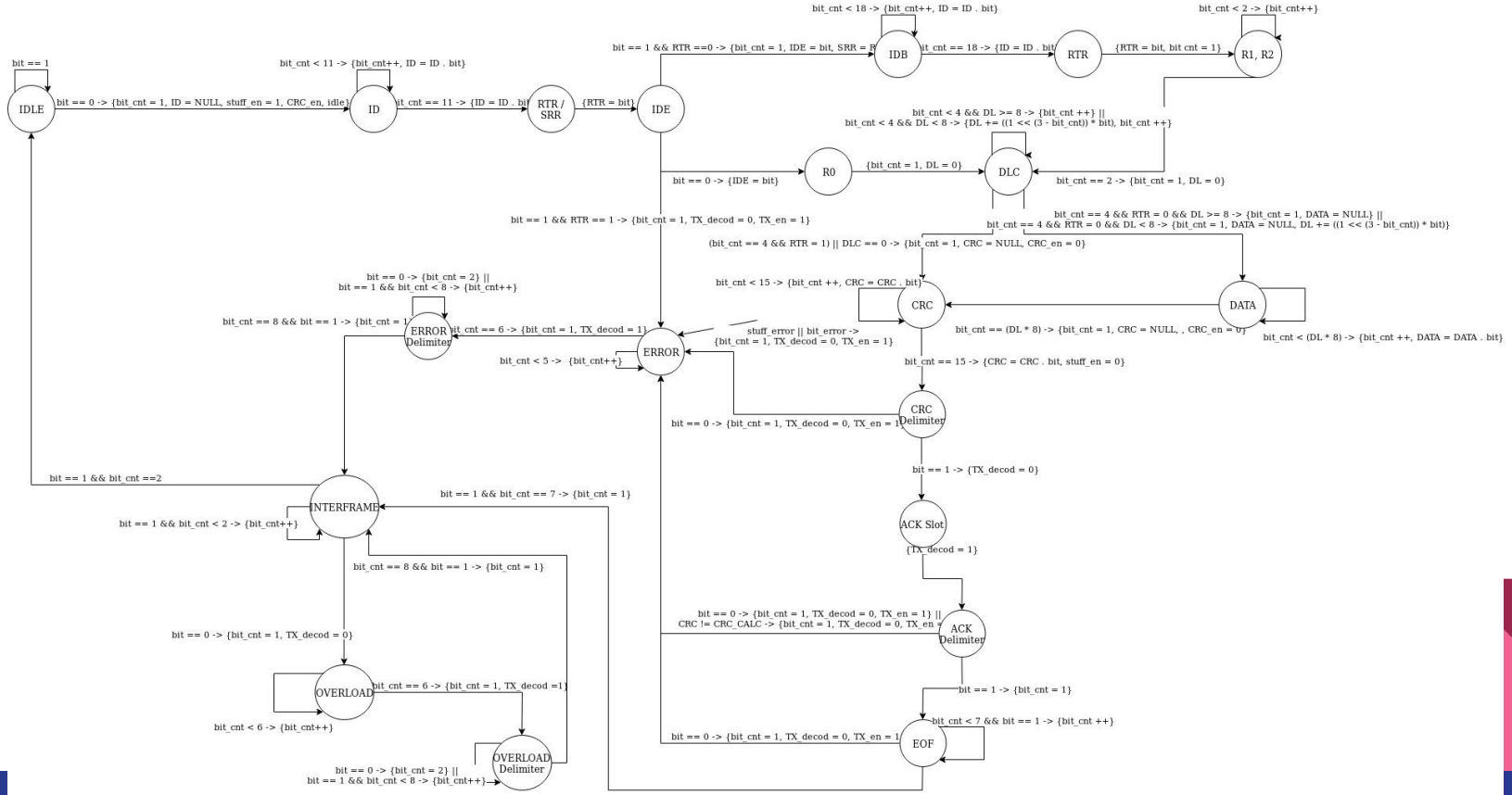
```
switch(state){
  case IDLE:…
  case SOF:…
  case ID:…
  case SRR:…
  case IDE:…
  case R0:…
  case IDB:…
  case RTR:…
  case R1:…
  case R2:…
  case DLC:…
  case DATA:…
  case CRC_V:…
  case CRC_D:…
  case ACK_S:…
  case ACK_D:…
  case EOFRAME:…
  case INTERFRAME:…
}
```

```
case R0:
  if(RX_bit != TX_bit) //BIT_ERROR
  {
    state = IDLE;
    TX_bit = 1;
    bit_error = 1;
    break;
  }
  TX_bit = (frame_send.DLC >> 3)&1;
  bit_cnt = 1;
  state = DLC;
  break;
```

Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_CALC
CRC_en
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

Decoder

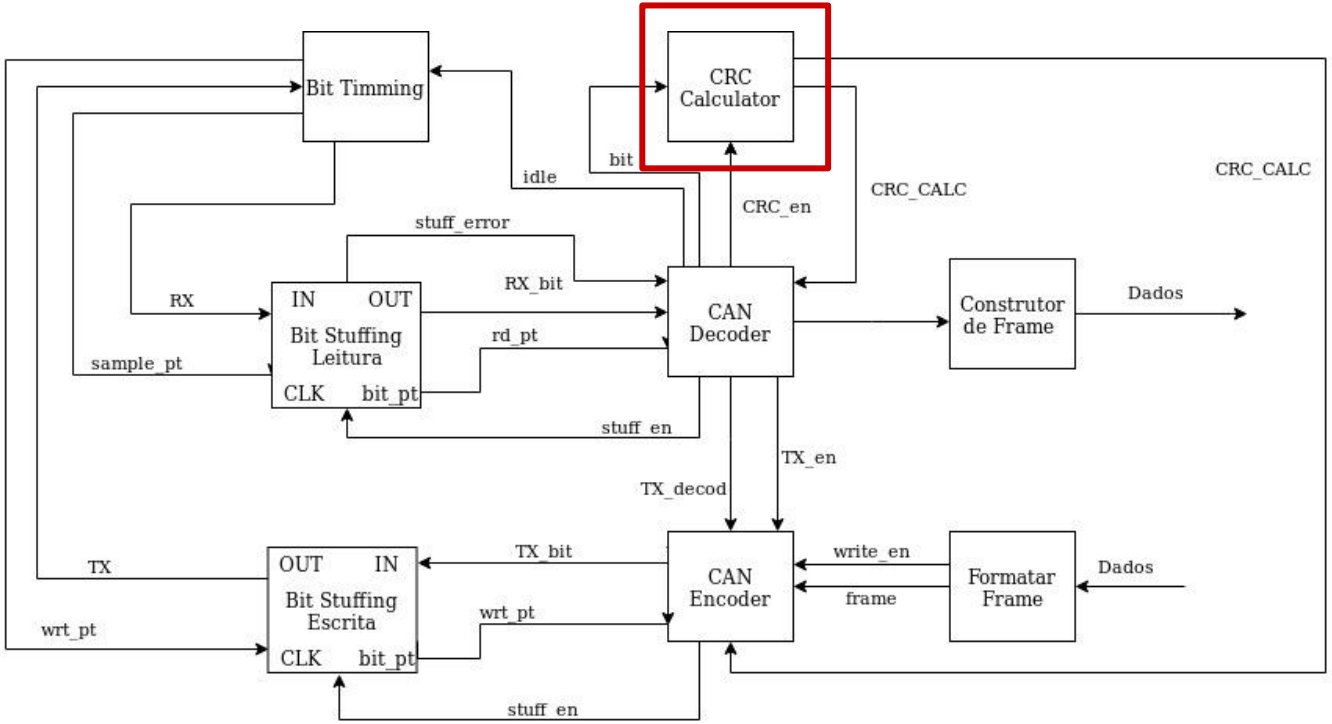

IDLE
bit == 1
bit == 0 -> {bit_cnt = 1, ID = NULL, stuff_en = 1, CRC_en, idle}
ID
bit_cnt < 11 -> {bit_cnt++, ID = ID . bit}
bit_cnt == 11 -> {ID = ID . bit}
RTR / SRR
{RTR = bit}
IDE
bit == 1 && RTR ==0 -> {bit_cnt = 1, IDE = bit, SRR = R
IDB
bit_cnt < 18 -> {bit_cnt++, ID = ID . bit}
bit_cnt == 18 -> {ID = ID . bit}
RTR
{RTR = bit, bit cnt = 1}
R1, R2
bit_cnt < 2 -> {bit_cnt++}
bit == 0 -> {IDE = bit}
R0
{bit_cnt = 1, DL = 0}
DLC
bit_cnt < 4 && DL >= 8 -> {bit_cnt ++} ||
bit_cnt < 4 && DL < 8 -> {DL += ((1 << (3 - bit_cnt)) * bit), bit_cnt ++}
bit_cnt == 2 -> {bit_cnt = 1, DL = 0}
bit == 1 && RTR == 1 -> {bit_cnt = 1, TX_decod = 0, TX_en = 1}
(bit_cnt == 4 && RTR = 1) || DLC == 0 -> {bit_cnt = 1, CRC = NULL, CRC_en = 0}
bit_cnt == 4 && RTR = 0 && DL >= 8 -> {bit_cnt = 1, DATA = NULL} ||
bit_cnt == 4 && RTR = 0 && DL < 8 -> {bit_cnt = 1, DATA = NULL, DL += ((1 << (3 - bit_cnt)) * bit)}
bit_cnt < 15 -> {bit_cnt ++, CRC = CRC . bit}
CRC
DATA
bit_cnt == (DL * 8) -> {bit_cnt = 1, CRC = NULL, , CRC_en = 0}
bit_cnt < (DL * 8) -> {bit_cnt ++, DATA = DATA . bit}
bit_cnt == 15 -> {CRC = CRC . bit, stuff_en = 0}
CRC Delimiter
bit == 0 -> {bit_cnt = 1, TX_decod = 0, TX_en = 1}
bit == 0 -> {bit_cnt = 2} ||
bit == 1 && bit_cnt < 8 -> {bit_cnt++}
bit_cnt == 8 && bit == 1 -> {bit_cnt = 1}
ERROR Delimiter
bit_cnt == 6 -> {bit_cnt = 1, TX_decod = 1}
stuff_error || bit_error ->
{bit_cnt = 1, TX_decod = 0, TX_en = 1}
ERROR
bit_cnt < 5 -> {bit_cnt++}
bit == 1 -> {TX_decod = 0}
ACK Slot
{TX_decod = 1}
bit == 1 && bit_cnt ==2
INTERFRAME
bit == 1 && bit_cnt == 7 -> {bit_cnt = 1}
bit == 1 && bit_cnt < 2 -> {bit_cnt++}
bit_cnt == 8 && bit == 1 -> {bit_cnt = 1}
bit == 0 -> {bit_cnt = 1, TX_decod = 0, TX_en = 1} ||
CRC != CRC_CALC -> {bit_cnt = 1, TX_decod = 0, TX_en =
ACK Delimiter
bit == 0 -> {bit_cnt = 1, TX_decod = 0}
OVERLOAD
bit_cnt == 6 -> {bit_cnt = 1, TX_decod =1}
bit_cnt < 6 -> {bit_cnt++}
bit == 1 -> {bit_cnt = 1}
bit_cnt < 7 && bit == 1 -> {bit_cnt ++}
EOF
bit == 0 -> {bit_cnt = 1, TX_decod = 0, TX_en = 1}
OVERLOAD Delimiter
bit == 0 -> {bit_cnt = 2} ||
bit == 1 && bit_cnt < 8 -> {bit_cnt++}

Decoder

```
void decoder(){
  static int state = 0;
  static int bit_cnt = 0;
  bool bit = RX_bit;
  if(stuff_error || bit_error)
  {…
  }
```

```
  switch(state)
  {
    case(IDLE):…
    case(ID):…
    case(SRR):…
    case(IDE):…
    case(R0):…
    case(IDB):…
    case(RTR):…
    case(R1):…
    case(R2):…
    case(DLC):…
    case(DATA):…
    case(CRC_V):…
    case(CRC_D):…
    case(ACK_S):…
    case(ACK_D):…
    case(EOFRAME):…
    case(INTERFRAME):…
    case(OVERLOAD):…
    case(OVERLOAD_D):…
    case(ERROR_FLAG):…
   case(ERROR_D):…
  }
  calculateCRC(bit);
}
```

Can Network

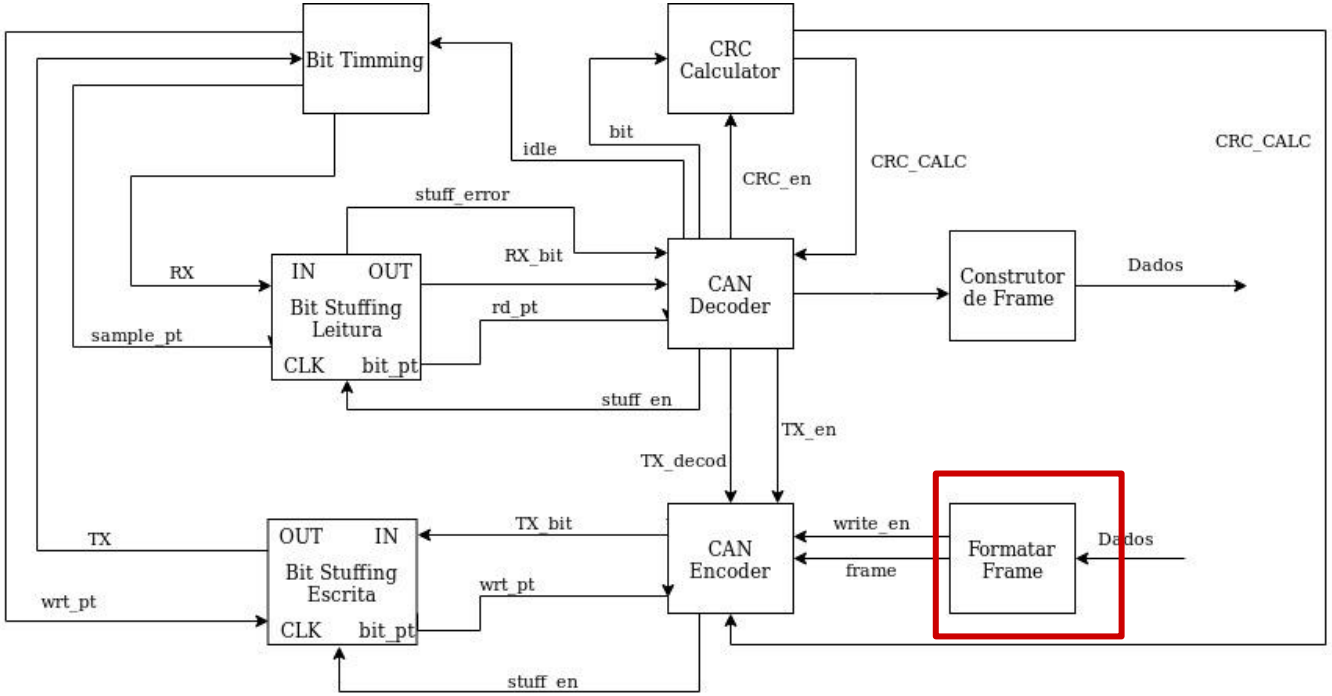

Bit Timming
CRC Calculator
bit
idle
CRC_en
CRC_CALC
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

CRC Calculator

```
CRC_CALC = [15 bits = 0];
while(CRC_en){
   CRC_first = bit ^ CRC_CALC[14];
   CRC_CALC << 1;
   if(CRC_first){
      CRC_CALC = CRC_CALC ^ hex(4599);
   }
}
```

```
void calculateCRC(bool bit)
{
  if (CRC_en) {
    CRC_CALC <<= 1;
    if ((CRC_CALC >= (1 << 15)) ^ bit) { // um smente no bit mais significativo
      CRC_CALC ^= 0x4599;
    }
    CRC_CALC &= 0x7fff; // zero no bit mais significativo e um no resto
  }
}
```

Can Network


Bit Timming
CRC Calculator
idle
bit
CRC_CALC
CRC_en
CRC_CALC
stuff_error
RX
IN
OUT
Bit Stuffing Leitura
CLK
bit_pt
RX_bit
rd_pt
sample_pt
CAN Decoder
Construtor de Frame
Dados
stuff_en
TX_en
TX_decod
TX
OUT
IN
Bit Stuffing Escrita
CLK
bit_pt
TX_bit
wrt_pt
wrt_pt
CAN Encoder
write_en
frame
Formatar Frame
Dados
stuff_en

Formatar Frame

- Struct used for frame parts.
- frame_receive filled by Decoder.
- frame_send read by Encoder.

```
typedef struct CAN_FRAME
{
  bool SOF;
  uint16_t ID;
  bool SRR;
  bool RTR;
  bool IDE;
  bool R0;
  uint32_t IDB;
  bool R1;
  bool R2;
  uint8_t DLC;
  uint64_t DATA;
  bool data_b = false;
  uint16_t CRC_V;
  bool CRC_D;
  bool ACK_S;
  bool ACK_D;
  uint8_t EOFRAME;
} CAN_FRAME;
```

```
frame_send.SOF = 0;
frame_send.ID = 0x20;
frame_send.SRR = 0;
frame_send.RTR = 0;
frame_send.IDE = 0;
frame_send.R0 = 0;
frame_send.IDB = 0;
frame_send.R1 = 0;
frame_send.R2 = 0;
frame_send.DLC = 2;
frame_send.DATA = 0xaaaa;
frame_send.data_b = false;
frame_send.CRC_V = 30547;
frame_send.CRC_D = 1;
frame_send.ACK_S = 0;
frame_send.ACK_D = 1;
frame_send.EOFRAME = 127;
```

- In Decoder

```
frame_recv.ID = (frame_recv.ID << 1) ^ bit;
```

- In Encoder

```
TX_bit = (frame_send.ID >> (10-bit_cnt)) & 1;
```

Testes Encoder with Decoder

```
Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   8
DATA:  aaaaaaaaaaaaaaaa
CRC_V: 81
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending......
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 8
st machine: DATA: aaaaaaaaaaaaaaaa
st machine: CRC_Value: 81
st machine: CRC_CALC: 81
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   0               0
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   8               8
DATA:  aaaaaaaaaaaaaaaa        aaaaaaaaaaaaaaaa
CRC_V: 81              81
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1
```

```
Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   7
DATA:  aaaaaaaaaaaaaa
CRC_V: 22941
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending......
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 7
st machine: DATA: aaaaaaaaaaaaaa
st machine: CRC_Value: 22941
st machine: CRC_CALC: 22941
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   0               0
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   7               7
DATA:  aaaaaaaaaaaaaa  aaaaaaaaaaaaaa
CRC_V: 22941           22941
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

```
Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   3
DATA:  aaaaaa
CRC_V: 9665
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 3
st machine: DATA: aaaaaa
st machine: CRC_Value: 9665
st machine: CRC_CALC: 9665
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   0               0
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   3               3
DATA:  aaaaaa    aaaaaa
CRC_V: 9665            9665
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

Testes Encoder with Decoder

```
Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   0
DATA:  0
CRC_V: 13013
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 0
st machine: CRC_Value: 13013
st machine: CRC_CALC: 13013
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   0               0
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   0               0
DATA:  0         0
CRC_V: 13013           13013
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

```
Printing Frame:
ID:    672
RTR:   1
IDE:   0
IDB:   0
SRR:   0
DLC:   0
DATA:  0
CRC_V: 16656
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 1
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 0
st machine: CRC_Value: 16656
st machine: CRC_CALC: 16656
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   1               1
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   0               0
DATA:  0         0
CRC_V: 16656           16656
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

```
Printing Frame:
ID:    672
RTR:   1
IDE:   0
IDB:   0
SRR:   0
DLC:   1
DATA:  0
CRC_V: 1161
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 1
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 1
st machine: CRC_Value: 1161
st machine: CRC_CALC: 1161
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    672             672
RTR:   1               1
IDE:   0               0
IDB:   0               0
SRR:   0               0
DLC:   1               1
DATA:  0         0
CRC_V: 1161            1161
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

Testes Encoder with Decoder

```
Printing Frame:
ID:    449
RTR:   0
IDE:   1
IDB:   3007a
SRR:   1
DLC:   8
DATA:  aaaaaaaaaaaaaaaa
CRC_V: 31733
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 449
st machine: RTR/SRR: 1
st machine: IDE: 1
st machine: SRR: 1
st machine: IDB: 3007a
st machine: RTR: 0
st machine: R1: 0
st machine: R2: 0
st machine: DLC: 8
st machine: DATA: aaaaaaaaaaaaaaaa
st machine: CRC_Value: 31733
st machine: CRC_CALC: 31733
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    449             449
RTR:   0               0
IDE:   1               1
IDB:   3007a           3007a
SRR:   1               1
DLC:   8               8
DATA:  aaaaaaaaaaaaaaaa        aaaaaaaaaaaaaaaa
CRC_V: 31733           31733
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

```
Printing Frame:
ID:    449
RTR:   1
IDE:   1
IDB:   3007a
SRR:   1
DLC:   8
DATA:  0
CRC_V: 10742
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 449
st machine: RTR/SRR: 1
st machine: IDE: 1
st machine: SRR: 1
st machine: IDB: 3007a
st machine: RTR: 1
st machine: R1: 0
st machine: R2: 0
st machine: DLC: 8
st machine: CRC_Value: 10742
st machine: CRC_CALC: 10742
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    449             449
RTR:   1               1
IDE:   1               1
IDB:   3007a           3007a
SRR:   1               1
DLC:   8               8
DATA:  0         0
CRC_V: 10742           10742
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

```
Printing Frame:
ID:    3
RTR:   0
IDE:   1
IDB:   0
SRR:   1
DLC:   15
DATA:  fffffffffffff
CRC_V: 20214
CRC_D: 1
ACK_S: 0
ACK_D: 1

Sending....
..
st machine: START OF FRAME
st machine: ID: 3
st machine: RTR/SRR: 1
st machine: IDE: 1
st machine: SRR: 1
st machine: IDB: 0
st machine: RTR: 0
st machine: R1: 0
st machine: R2: 0
st machine: DLC: 15 --> DLC: 8
st machine: DATA: fffffffffffff
st machine: CRC_Value: 20214
st machine: CRC_CALC: 20214
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frames:
       Sent:           Receive:
ID:    3               3
RTR:   0               0
IDE:   1               1
IDB:   0               0
SRR:   1               1
DLC:   15              8
DATA:  fffffffffffff   fffffffffffff
CRC_V: 20214           20214
CRC_D: 1               1
ACK_S: 0               0
ACK_D: 1               1

st machine: IDLE
```

Testes Encoder with Decoder Repeating

```
Sending....

Printing Frames:
        Sent:           Receive:
ID:     672             672
RTR:    0               0
IDE:    0               0
IDB:    0               0
SRR:    0               0
DLC:    3               3
DATA:   aaaaaa    aaaaaa
CRC_V: 9665             9665
CRC_D: 1                1
ACK_S: 0                0
ACK_D: 1                1

Sending....

Printing Frames:
        Sent:           Receive:
ID:     672             672
RTR:    0               0
IDE:    0               0
IDB:    0               0
SRR:    0               0
DLC:    3               3
DATA:   aaaaaa    aaaaaa
CRC_V: 9665             9665
CRC_D: 1                1
ACK_S: 0                0
ACK_D: 1                1
```

```
Sending....

Printing Frames:
        Sent:           Receive:
ID:     672             672
RTR:    0               0
IDE:    0               0
IDB:    0               0
SRR:    0               0
DLC:    3               3
DATA:   aaaaaa    aaaaaa
CRC_V: 9665             9665
CRC_D: 1                1
ACK_S: 0                0
ACK_D: 1                1

Sending....

Printing Frames:
        Sent:           Receive:
ID:     672             672
RTR:    0               0
IDE:    0               0
IDB:    0               0
SRR:    0               0
DLC:    3               3
DATA:   aaaaaa    aaaaaa
CRC_V: 9665             9665
CRC_D: 1                1
ACK_S: 0                0
ACK_D: 1                1
```

Testes Decoder Repeating Frame

```
................................................................................
st machine: START OF FRAME
st machine: ID: 3
st machine: RTR/SRR: 1
st machine: IDE: 1
st machine: SRR: 1
st machine: IDB: 0
st machine: RTR: 0
st machine: R1: 0
st machine: R2: 0
st machine: DLC: 15 --> DLC: 8
st machine: DATA: ffffffffffffffff
st machine: CRC_Value: 11311
st machine: CRC_CALC: 11311
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME

Printing Frame:
ID:    3
RTR:   0
IDE:   1
IDB:   0
SRR:   1
DLC:   8
DATA:  ffffffffffffffff
CRC_V: 11311
CRC_D: 1
ACK_S: 0
ACK_D: 1

st machine: IDLE
................................................................................
st machine: START OF FRAME
st machine: ID: 3
st machine: RTR/SRR: 1
st machine: IDE: 1
st machine: SRR: 1
st machine: IDB: 0
st machine: RTR: 0
st machine: R1: 0
st machine: R2: 0
st machine: DLC: 15 --> DLC: 8
st machine: DATA: ffffffffffffffff
st machine: CRC_Value: 11311
st machine: CRC_CALC: 11311
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: EOFRAME
st machine: INTERFRAME
```

Testes Decoder with Error

```
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 8
st machine: DATA: aaaaaaaaaaaaaaaa
st machine: CRC_Value: 81
st machine: CRC_CALC: 81
st machine: CRC_D: 1
st machine: ACK_S: 1
st machine: ACK_D: 0
st machine: ACK Error
st machine: ERROR_FLAG
st machine: ERROR_DELIMITER
st machine: INTERFRAME

Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   8
DATA:  aaaaaaaaaaaaaaaa
CRC_V: 81
CRC_D: 1
ACK_S: 1
ACK_D: 0

st machine: IDLE
```

```
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 8
st machine: DATA: aaaaaaaaaaaaaaaa
st machine: Error Detected: STUFF_ERROR
st machine: ERROR_FLAG
st machine: ERROR_DELIMITER
st machine: INTERFRAME

Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   8
DATA:  aaaaaaaaaaaaaaaa
CRC_V: 0
CRC_D: 0
ACK_S: 0
ACK_D: 0

st machine: IDLE
```

```
st machine: START OF FRAME
st machine: ID: 672
st machine: RTR/SRR: 0
st machine: IDE: 0
st machine: R0: 0
st machine: DLC: 8
st machine: DATA: aaaaaaaaaaaaaaaa
st machine: CRC_Value: 113
st machine: CRC_CALC: 81
st machine: CRC_D: 1
st machine: ACK_S: 0
st machine: ACK_D: 1
st machine: CRC_V =! CRC_CALC
st machine: ERROR_FLAG
st machine: ERROR_DELIMITER
st machine: INTERFRAME

Printing Frame:
ID:    672
RTR:   0
IDE:   0
IDB:   0
SRR:   0
DLC:   8
DATA:  aaaaaaaaaaaaaaaa
CRC_V: 113
CRC_D: 1
ACK_S: 0
ACK_D: 1

st machine: IDLE
```

Working Schedule

	Felipe Martins	Lucas Cavalcanti	Roberto Fernandes
Bit Timing			
Bit Stuffing			
Encoder			
Decoder			
Integration			
Validation			
Reports			

Problems Found

- Transceiver not working.
- Implementation in a single core microcontroller.
- No ground truth implemented.
- Debugging interface is slow.
- Microcontroller does not support 500Kbps CAN Network.
- Implementation of state machines need to respect the priorities.
- One state machine interfere in others.

Conclusion

- CAN Network should be implemented in FPGA.
- Implementing CAN Network in C/C++ improves its comprehension.
- CAN Network has an intelligent strategy for synchronizing.