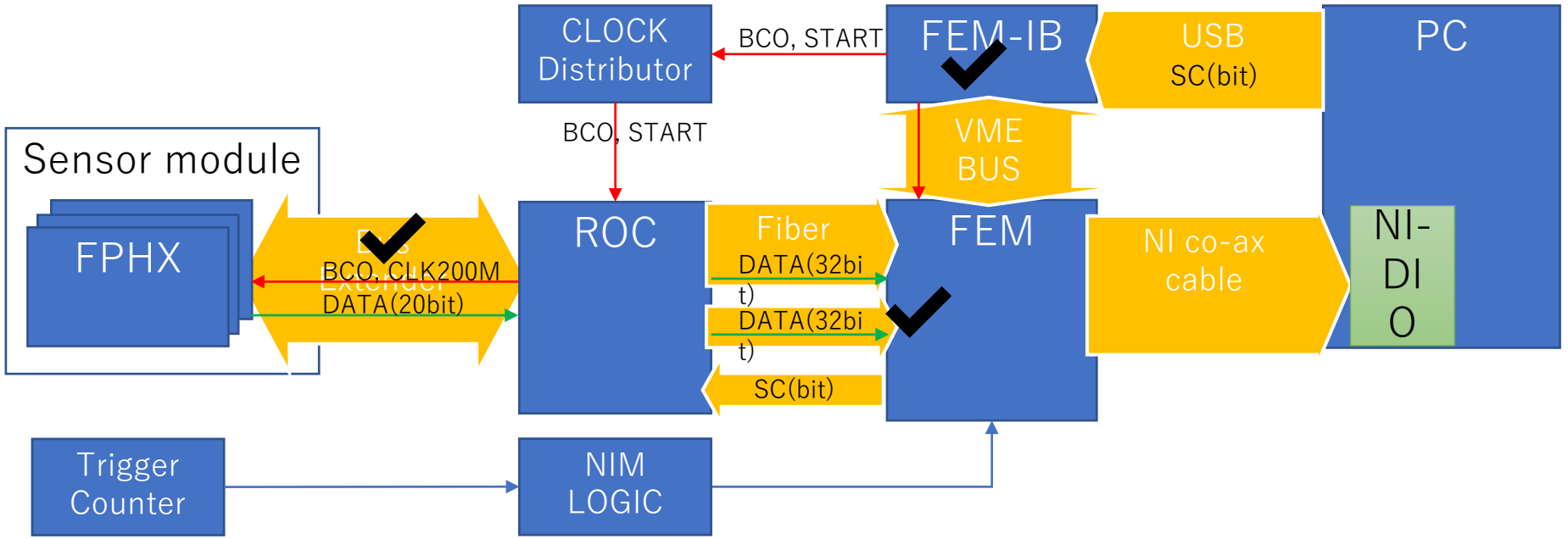
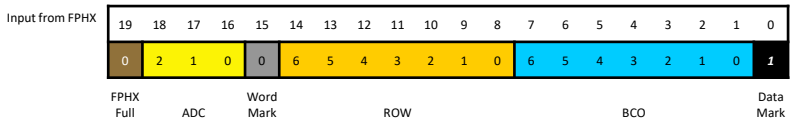
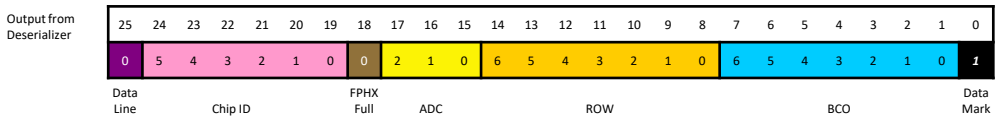
Readbackerの状況

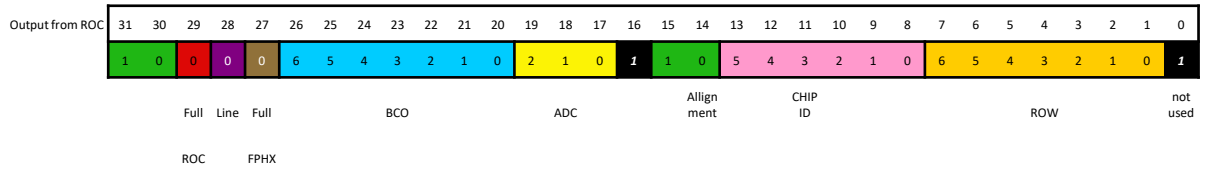
蜂谷　崇、高濱　瑠菜

INTT readout system @ Test Bench

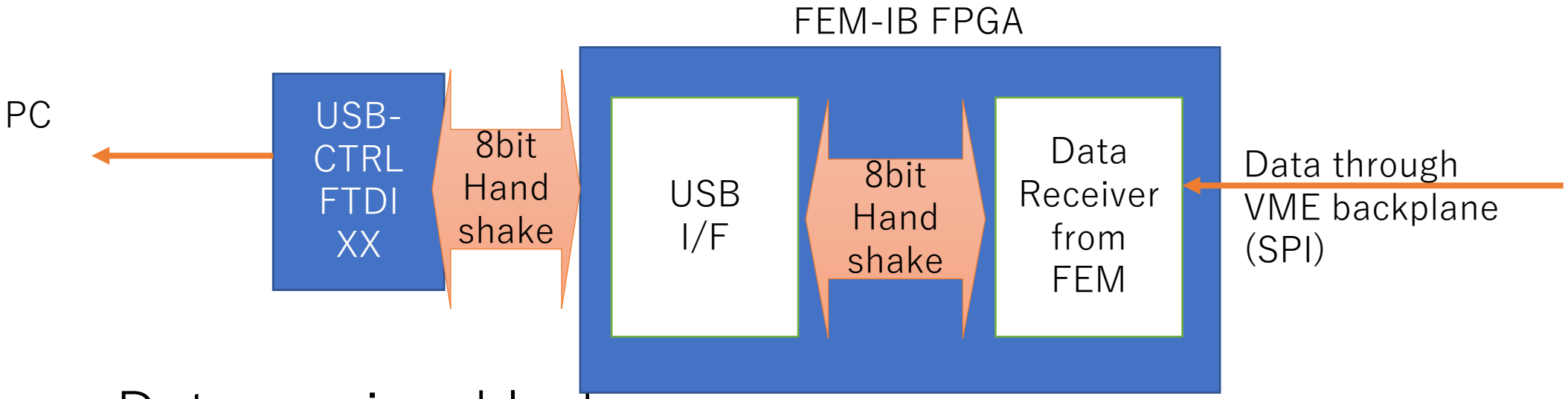








What is needed for FEM-IB codes and DAQ-GUI



- Data receiver block
  - テストベンチ版にモジュールがあったので、中身を改造して使う。
- USB I/F block
  - この部分は無いので、簡易版をつくる。
- Handshake
  - レジスタに保存しておく。

Data receiver block

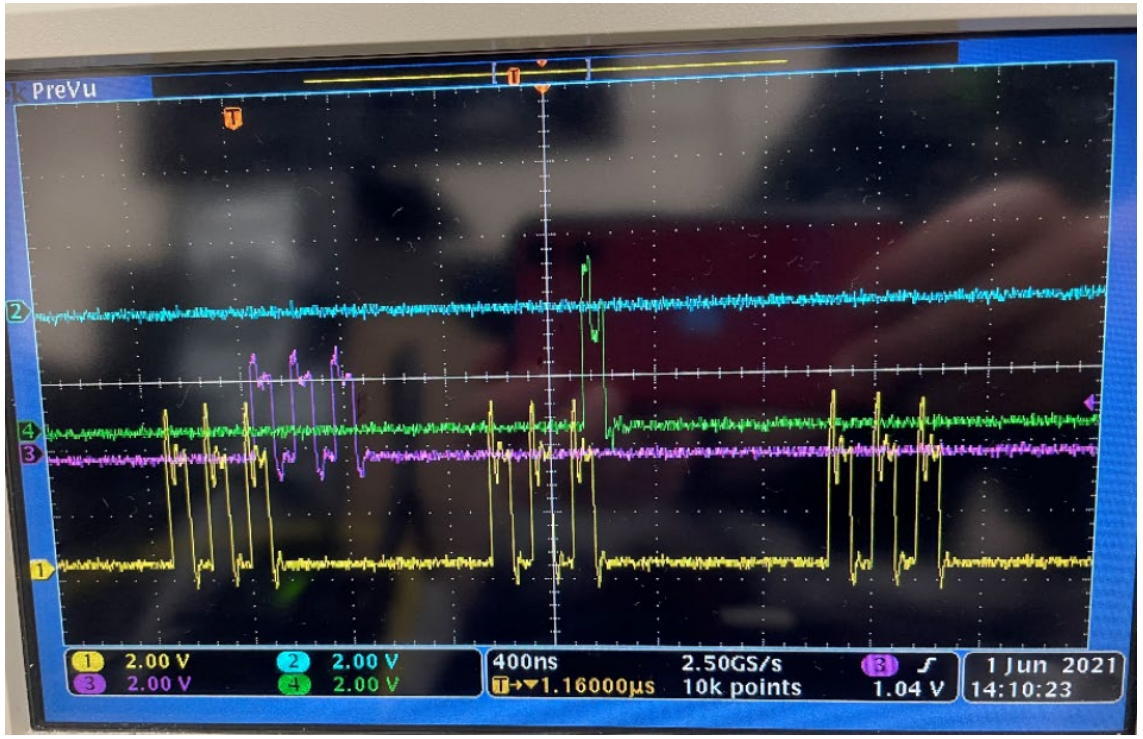
- 16bit データとしてFEM-IBに戻ってくる
  - MSB-8bit : 固定のヘッダー 0b_10001101
  - LSB-8bit ：ReadBackしたデータ

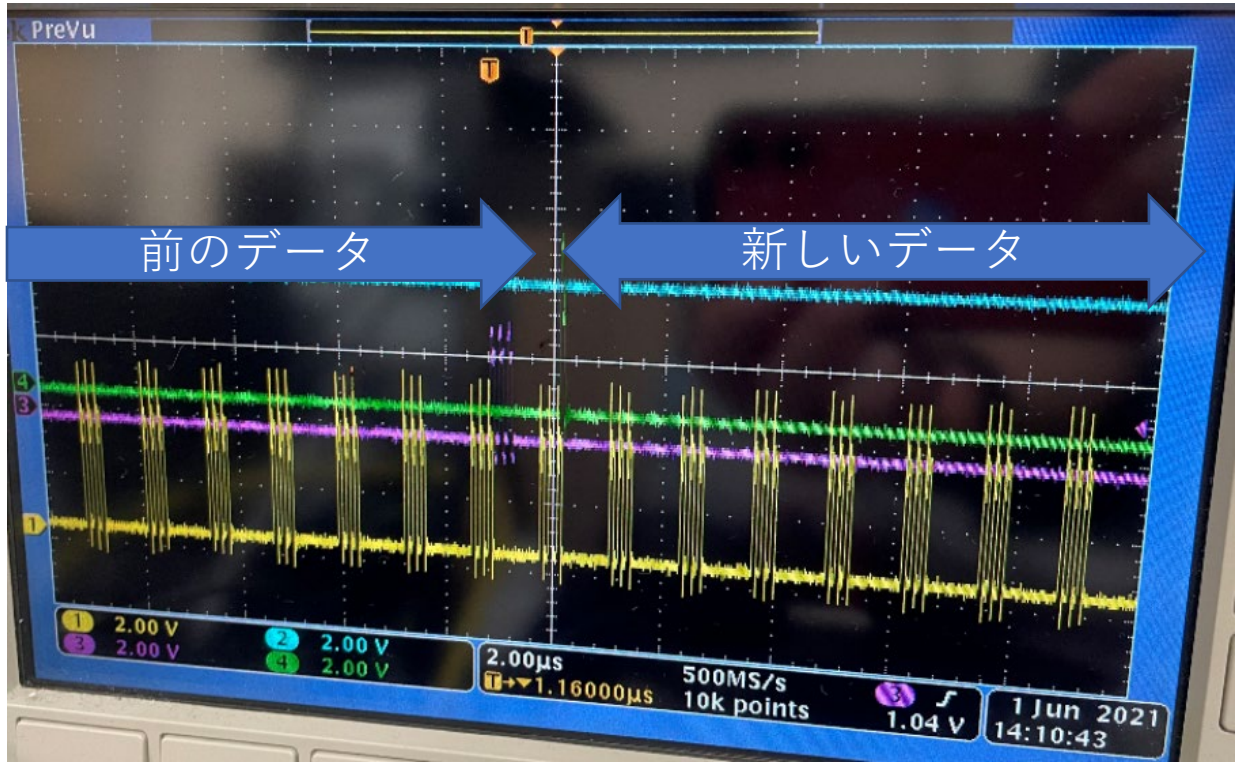
- 既存のData Receiver blockの機能
  - **ReadBackデータをSerial → Parallel 変換**
    - Readbackヘッダーの受信で動作開始
  - VMEへの8ビットデータ(コマンド)を保存

- 追加
  - 最新のReadbackデータを常時保存
    - 新しいのが来たら書き換える

受信データの保持

同じデータを2度読み出した





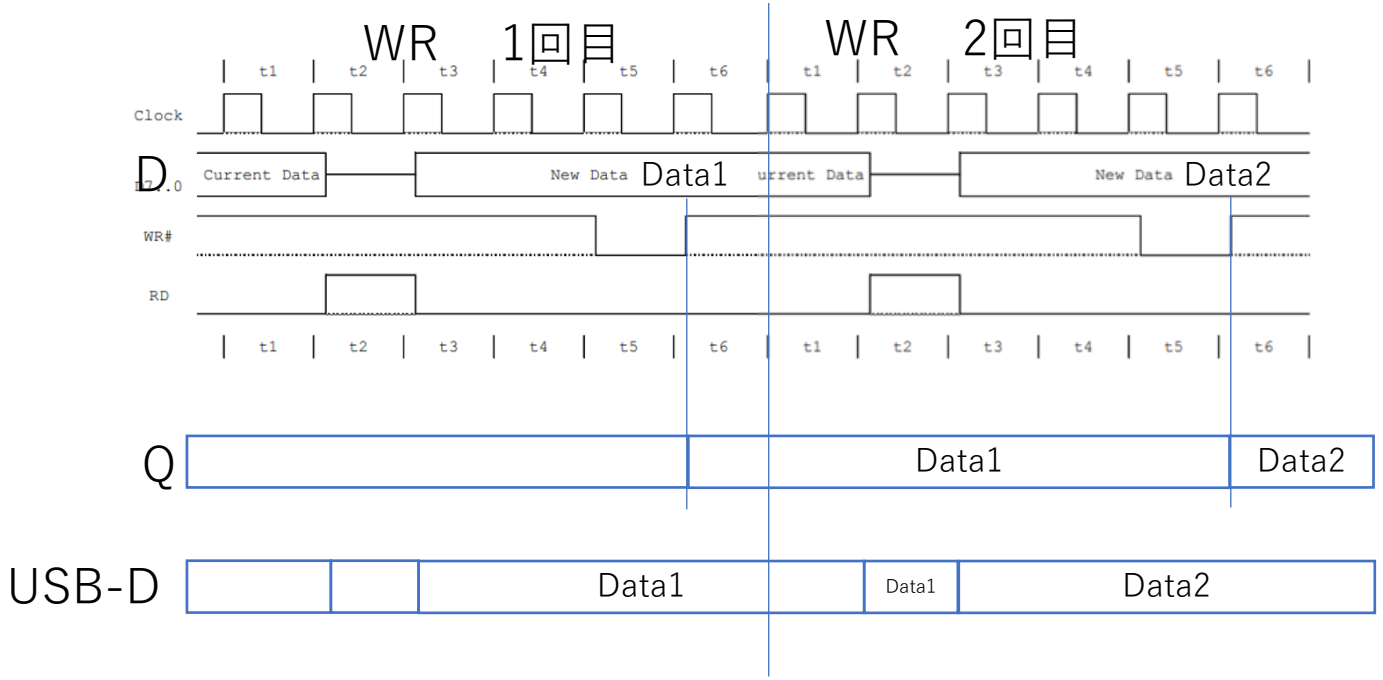
FEMからの出力(Readbackデータ)
FEMIBでHeader確認
保持後のデータ

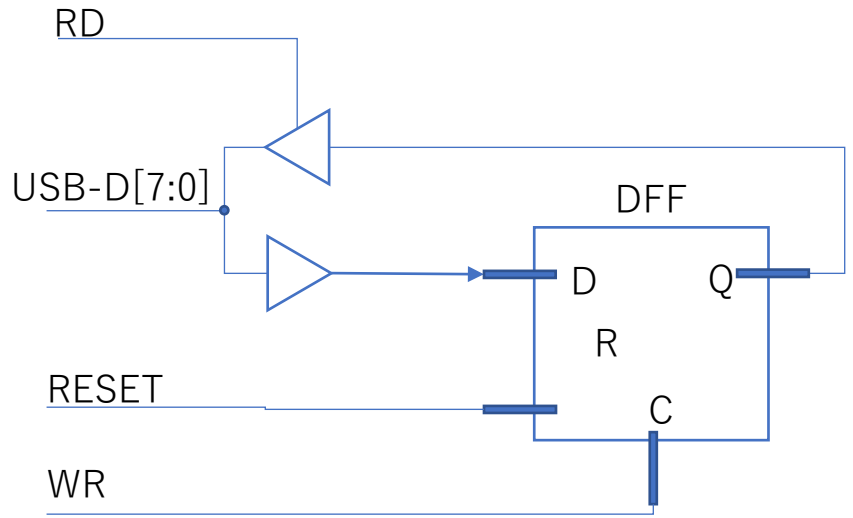
- 読み出したデータをずっと保持できている。

USB－I/Fで課題

- 読み出しに成功したと思っていたが、そうではなさそう。

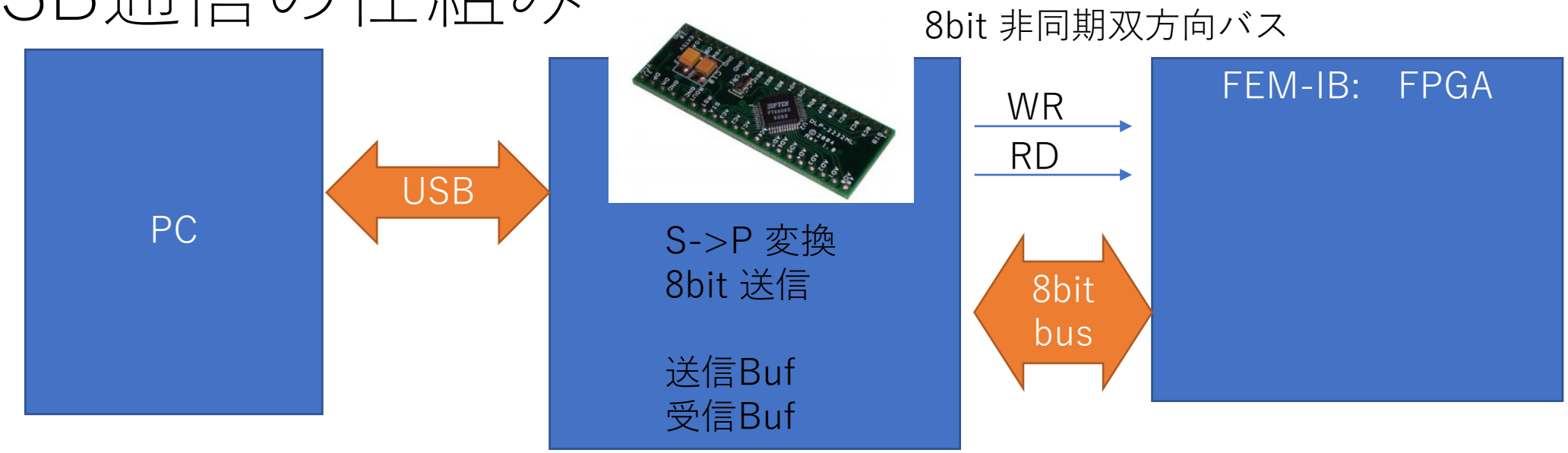
入力したデータ(8bit)をそのまま返す回路　→　返ってきたのでOKと思っていた。
次にコマンド受け付け数(カウンタ)を返すように変更　→　返ってこず、　データが返ってきた。



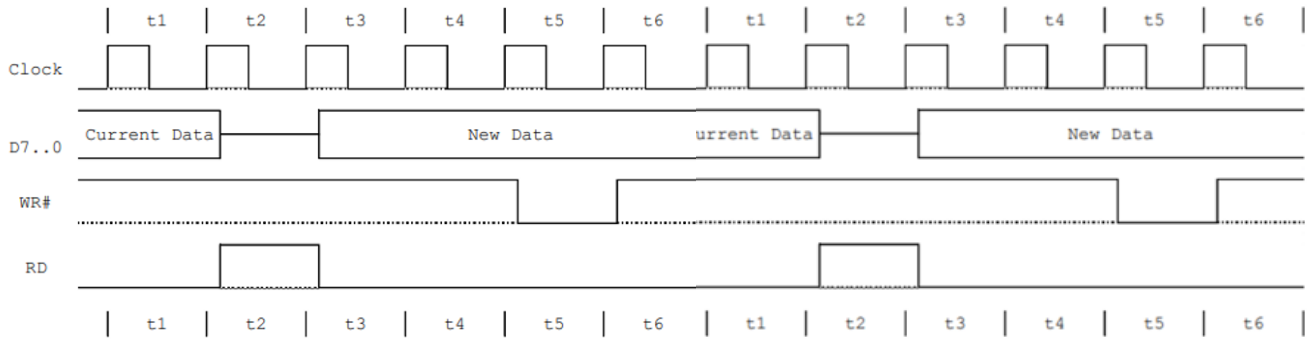


読み出しに成功したと思っていたが、そうではなさそう。

USB通信の仕組み

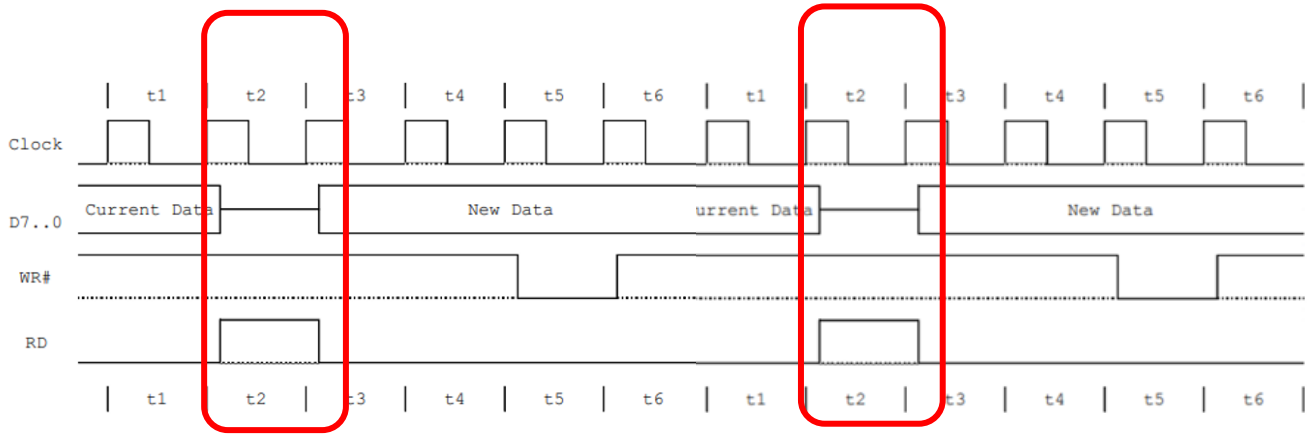


- USBからの送り方：同期BitBangモード
  - 8bitワードを連続して送る
  - 1ワード送るたびに受信もする。
    - 受信　→　送信の順



- PCからのIO
  - Unsigned char *wrbuf = …
  - Write(wrbuf, size)
    - Size ワード分、連続して書き込む
  - Read(rdbuf, size)
    - Size ワード分、読み出す。　読み出し終わるまで待つ

今の考察



- 観測
  - 送ったコマンドは読み出せる
  - カウンタは読み出せない

- 起こっていることの予想
  - BUSの方向切り替えがうまくいっていない。
    - DATA線に残っている、送信データを読み込んでいるのではないか。

- 8Bitのデータラインの状態を読み出してみる。

```
entity receive_block is
    Port ( DATA_IN : in std_logic;
           CS_SC_VME : in std_logic;
           COMMAND_VME : in std_logic_vector(7 downto 0);
           CLK : in std_logic;
           RST : in std_logic;
           DATA_OUT : out std_logic_vector(7 downto 0));
end receive_block;

architecture RTL of receive_block is

signal  SH_REG : std_logic_vector (15 downto 0);
signal CS_SC_VME_BUF : std_logic;
signal CS_SC_VME_BUF2 : std_logic;

begin

   process (CLK, RST)
   begin
      if RST = '1' then
         SH_REG <= (others => '0');
         DATA_OUT <= (others => '0');
         CS_SC_VME_BUF <= '0';
         CS_SC_VME_BUF2 <= '0';
      --FEM sends packet on rising edge so we should check on falling edge:
      --elsif rising_edge(CLK) then
      elsif falling_edge(CLK) then
         --Set the status response to be COMMAND any time a new command is sent.  If a
         --valid status response packet is received after this, it will get overwritten
         --with the status packet:
         CS_SC_VME_BUF <= CS_SC_VME;
         CS_SC_VME_BUF2 <= CS_SC_VME_BUF;
         if (CS_SC_VME_BUF = '1' and CS_SC_VME_BUF2 = '0') then
            DATA_OUT <= COMMAND_VME;
         end if;

         SH_REG <= SH_REG(14 downto 0) & DATA_IN;
         if SH_REG(15 downto 8) = "10001101" then
            DATA_OUT <= SH_REG(7 downto 0);
         end if;
      end if;
   end process;


end RTL;
```