Привожу функцию открытия карты, чтобы ты почитал тут про память

```
int OpenCard(int SN)
{
        int i;
        i=pcicarrier_OpenCard(&DevIndex,  SN, &Memcs2, &Memcs3);
// Инициализирует драйвер и плату по ее серийному номеру CardSerNum.
// Возвращает 0-если успешно, 1-ошибка.
// Возвращает следующие переменные:
// DevIndex - индекс платы (0-3). Все остальные фунции используют этот индекс для работы
// memcs2 - указатель на 256 кб  8-бит памяти, выбираемой сигналом CS2.
// memcs3 - указатель на 256 кб 16-бит памяти, выбираемой сигналом CS3.
// С этими массивами можно работать в программе, как с обычными данными, обращение к нему
будет  приводить к генерации 8 или 16-бит обмена по CS2 или CS3.
// Внимание! Установка переменной из этих областей в окне watch приведет к постоянным
генерациям  опроса памяти при перерисовке окна.
        if (i)
        {
                return 1;
        }
        BMemcs2=(char *)Memcs2;
        BMemcs3=(short int *)Memcs3;
        return 0;
}
```

Функция инициализации TDC

```
int  InitTDC(void)
{
        int ErrorVME,res;
        unsigned short dd;

        //RESET
        //ЗАПРЕТ СТАРТА ,СТОПОВ и инициализации  ТДС,СБРОС FIFO
        //D7  = 0 - disable INTR from Flag FIFO
        // D6...D0  - xx
        Address=0;
        Value=0;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        // D6 -   0 - reset TDC;
        // D5 -   0 - reset FIFO(clear EF HF FF- 100);
        // D7,D4..D0 -xx
        Address=2;
        Value=0;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        //D15-D12 = 0x0 - disable STOP (T3,T2,T1,T0)
        //D11-D10 = 0   - disable START,
        //D9..D2 - xx
        //D1 = 0 - УСТАНОВКА В НАЧАЛЬНОЕ СОСТОЯНИЕ, СХЕМЫ ПРИЕМА СТОПОВ
        //D0 = 0 - RESET  TDC,!! З А П Р Е Т/С Б Р О С ИНИЦИАЛИЗАЦИИ TDC
        BMemcs3[31]=0;
```

```
        // ЗАПРЕТ ПРЕРЫВАНИЙ,СТРОПОВ и СТАРТА;РАЗРЕШЕНИЕ TDC и  FIFO
        //D5 = 1-enable WORK
        //    D6 = 1-enable TDC;
        Address=2;
        Value=0x40;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        //D1 = 1  РАЗРЕШЕНИЕ СХЕМЫ ПРИЕМА СТОПОВ
        //  D0 = 1 -  РАЗРЕШЕНИЕ ИНИЦИАЛИЗАЦИИ TDC
        BMemcs3[31]=0x3;

        //INIT TDC
        BMemcs3[1]=0x007; //rising edges, Start ringoscil.
        BMemcs3[0]=0xfc81;
        BMemcs3[3]=0;
        BMemcs3[2]=0;
        BMemcs3[5]=0;//I-mode
        BMemcs3[4]=2;
        BMemcs3[7]=0;
        BMemcs3[6]=0;
        BMemcs3[9]=0x600;//Mtimes trig, by Sart, EFlagHiZN
        BMemcs3[8]=0;
        BMemcs3[11]=0x0E0;  //StopDisStart, StartOff1 = 0, MasterAluTrig
        BMemcs3[10]=0x0;
        BMemcs3[13]=0;
        BMemcs3[12]=0;
        BMemcs3[15]=0x014; // 14 time meas = 1 mksec
        BMemcs3[14]=0x1FB4; // Res = 82,3045 ps
        BMemcs3[23]=0x7FF; //Any error -> ErrFlag
        BMemcs3[22]=0;
        BMemcs3[25]=0x200; //Mtimer -> IrFlag
        BMemcs3[24]=0;
        BMemcs3[29]=0;
        BMemcs3[28]=0;
        BMemcs3[9]=0x640; // Master reset
        BMemcs3[8]=0;
        return 0;
}
```

Старт работы TDC
Int StartTDC(void)

```
{
        // D7  = 1 - enable INTR from Flag FIFO
        Address=0;
        Value=0x80;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        //RESET FIFO,ENABLE TDC
        // D5 =  0 - reset FIFO(clear EF HF FF- 100);
        // D6 =  1 - enable TDC
        Address=2;
        Value=0x40;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        //УСТАНОВКА В НАЧАЛЬНОЕ СОСТОЯНИЕ, СХЕМЫ ПРИЕМА СТОПОВ 1-РАЗРЕШЕНИЕ
        //  ENABLE TDC ,DISABLE START,STOPS
        // D15-D12 = 0xf          - enable STOP (3,2,1,0)
        // D11-D10       = 0   - disable START,
        // D1 = 0 - УСТАНОВКА В НАЧАЛЬНОЕ СОСТОЯНИЕ, СХЕМЫ ПРИЕМА СТОПОВ
        // D0 = 1 - РАЗРЕШЕНИЕ ИНИЦИАЛИЗАЦИИ TDC
        BMemcs3[31]=0xf001;

        //ENABLE FIFO,TDC
        //D5 = 1-enable WORK
        // D6 = 1-enable TDC;
        Address=2;
        Value=0x60;
        pcicarrier_WritePortCS0(DevIndex, Address, Value);

        //ENABLE START and STOPS,TDC
        // D15-D12 = 0xf          - enable STOP (3,2,1,0)
        // D11-D10       = 0x3 - enable START
        // D1 = 1 - РАЗРЕШЕНИЕ          СХЕМЫ ПРИЕМА СТОПОВ
        // D0 = 1 - РАЗРЕШЕНИЕ ИНИЦИАЛИЗАЦИИ TDC
        BMemcs3[31]=0xfc03;

        Return 0;
}
```

Останов работыTDC

Int SopTDC(void)

```
{
        //ENABLE START and STOPS,TDC
        // D15-D12 = 0xf          - enable STOP (3,2,1,0)
        // D11-D10       = 0x10 - ???
        // D1 = 1 - РАЗРЕШЕНИЕ          СХЕМЫ ПРИЕМА СТОПОВ
        // D0 = 1 - РАЗРЕШЕНИЕ ИНИЦИАЛИЗАЦИИ TDC
         BMemcs3[31]=0xf803;
Return 0;
}
```

Проверка заполненности памяти

```
int CheckInteruptFIFO(void )
{
        Address=3;
        Value= pcicarrier_ReadPortCS0( DevIndex, Address);
        return(Value & 3);
        //result=0 - память не заполнена
        //result=2 - память заполнена на половину
        //result=1 - память заполнена полностью

}
```

Чтение четверки данных из памяти

Все слова из памяти ФИФО читаются по этому адресу
DataFIFO[n] = BMemcs3[256];
Данные 16 разрядов такой структуры

	16 15 14 разряды	13 – 1 разряды
Порядок появления	Номер канала	Код времени
1	3	X1
2	1	Y1
3	5	Y2
4	7	X2

Если четверка не найдена, то данные прочитаны. Но бывает, что возникает лишнее слово. Его нужно игнорировать.