IOC for upgrading BPM DAQ software

WANG,Jian (王坚)

Univ. of Sci. & Tech. of China

IOC for upgrading BPM DAQ software

- IOC on win32
  - Various technologies of win32
  - Test IOC based on IVI-COM and TekVisa
  - Test IOC performance and common win32 application
- IOC for upgrading BPM DAQ software
  - DAQ IOC software
  - CA Client of DAQ IOC software

1. IOC on win32

- Different C/C++ compiler with different host platforms
  - Here use win32-x86: MS compiler
- Different products used the different MS technologies
  - ActiveX(ActiveDSO in wavePro oscilloscope), COM(XStream in wavePro oscilloscope, IVI-COM in Tektronix oscilloscope), TekVisa (Tektronix oscilloscope)
  - DLL could be used to integrate to IOC when writing a device support

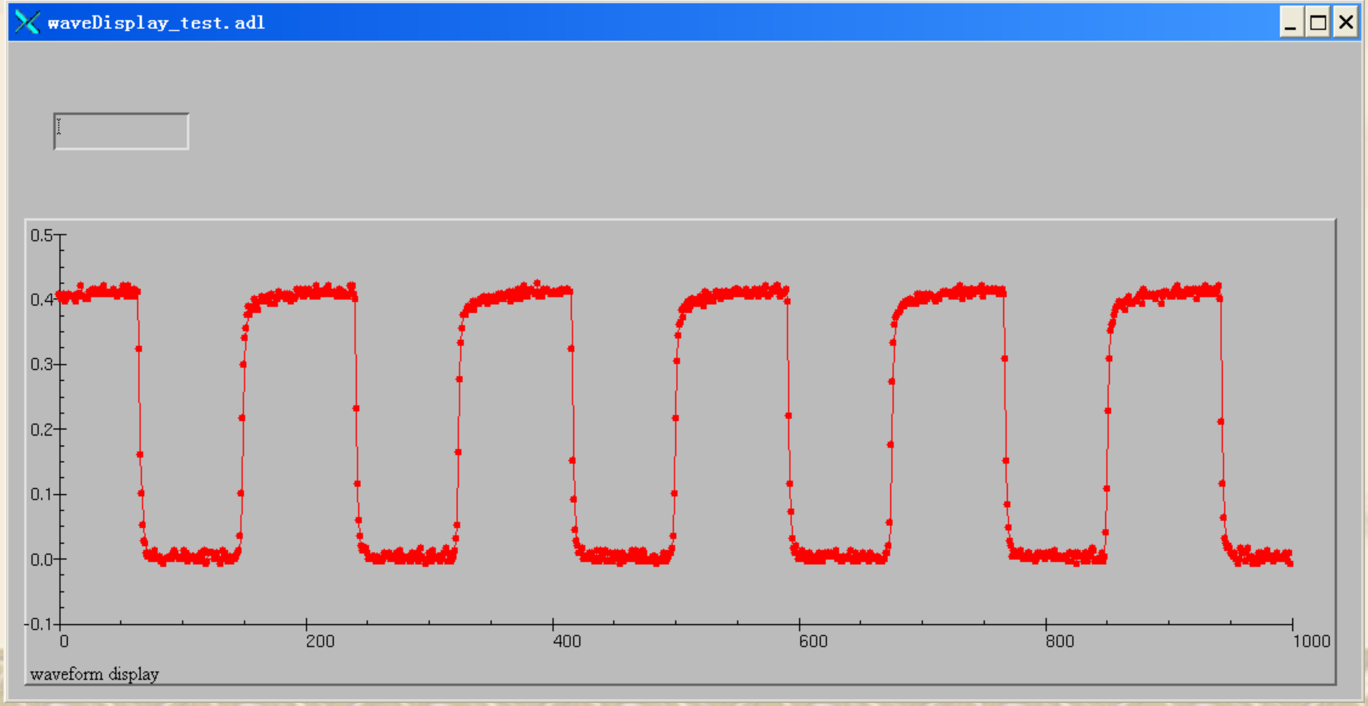
1.1 Test IOC based on IVI-COM and TekVisa

- Tek DPO 7104: Win XP
- EPICS IOC:base-3.14.8.2,VC2005,cygwin (gnu make, perl)
- Two methods for IOC
  - IVI-COM: Interchangeable Virtual Instruments
    - seems memory did not released when one scan was done(with ITekScopeWaveformTransfer.FetchWaveform Method)
  - TekVisa: as normal programming, add the tekvisa library directory to lib path and add include directory to include path in the makefile as follows
    - USR_INCLUDES += -I"C:¥VXIpnp¥WINNT¥include“
    - xxxSupport_SYS_LIBS += visa32

IVI-COM

- IVI: Interchangeable Virtual Instruments
- Installed IVI TekScope Driver firstly
- In device support, C++ must be used for COM technology
- Code like follows
  - ::OleInitialize(NULL); //before create instance
  - …   //reference IVI-COM sample
  - ::OleUninitialize(); // when done

- Get waveform
  - IOC running on OSC
  - Medm remotely



- Compare IVI-COM and TekVisa
  - The IOC used IVI-COM running as scan periodic
    - The memory will be exhausted and at last it will be ended with an error
  - The fastest speed of acquisition is lower than that using TekVisa
- So IVI-COM is unselected

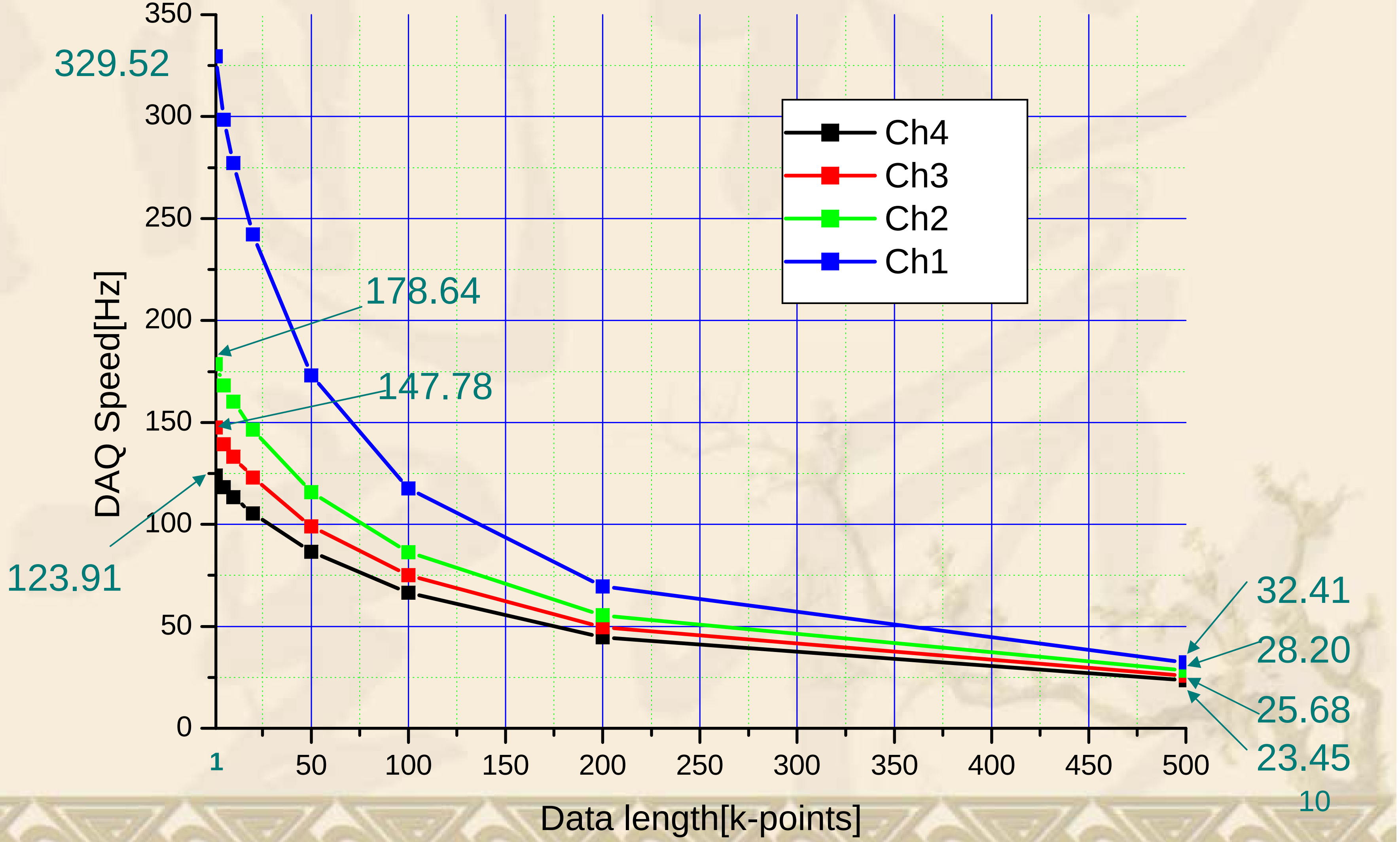
1.2 Test IOC performance and common win32 application

- Develop an IOC based on TekVisa
  - Waveform acquisition only
  - Vary the record length of waveform from 1000 to 500,000
  - Vary the scan periodic cycle and scan passive
    - The minimal scan period could be 0.01 second
    - Add 0.01s,0.02s,0.05s to scan menu type
  - Get the waveform continuously to get the average of acquisition speed
    - Amount of waveform acquisition >= 1000
- A tekVisa test program have developed using VC++ 2005

Test IOC performance and common win32 application

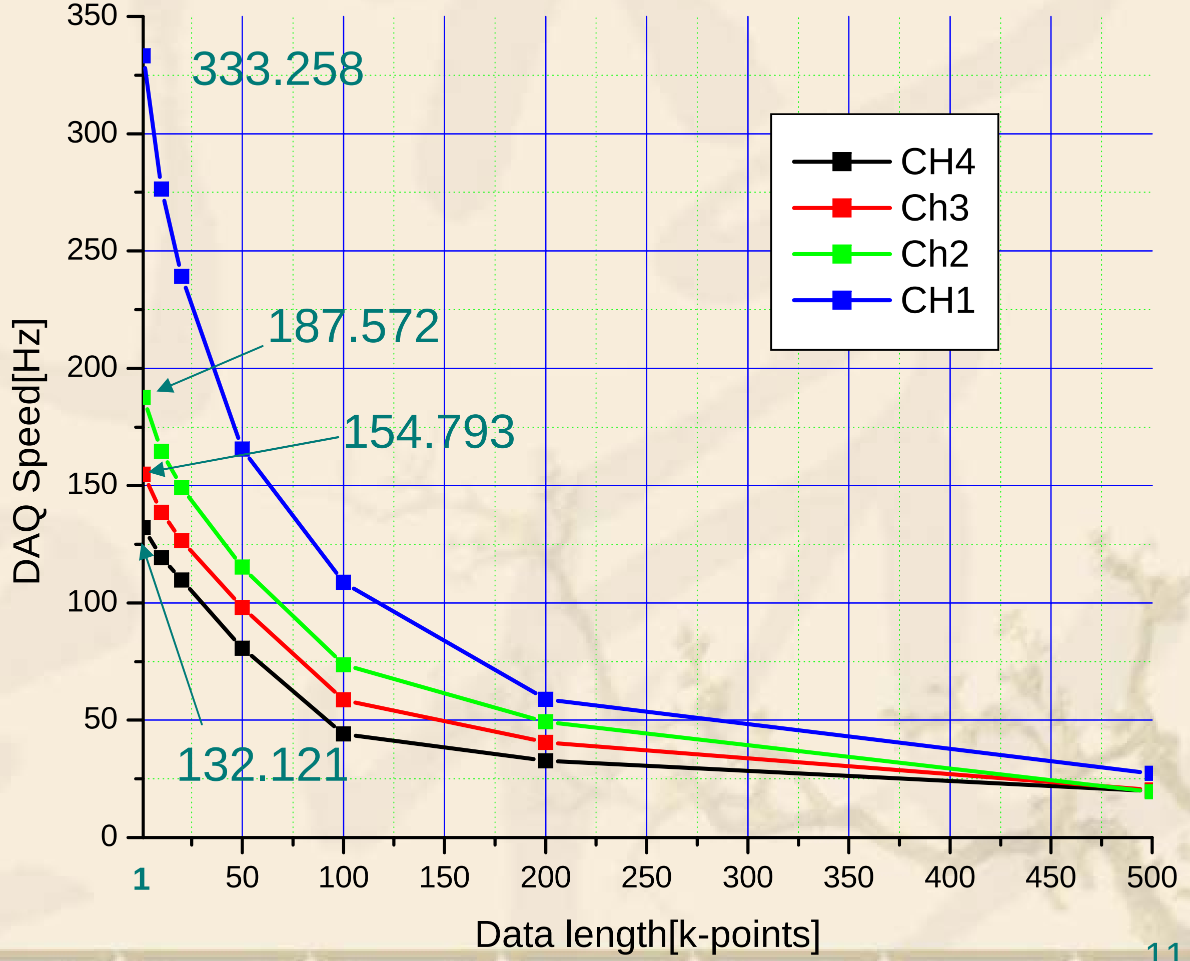
- Curve and curvestream
  - Curvestream make OSC to continuously transfer waveform data as fast as it is acquired. Also it puts instrument to a talk-only mode and no response to other clients and other commands.
  - Need time to stop curvestream mode to place OSC back into its normal talk/listen mode
  - So if settings of OSC seldom are reconfigured and fast performance is needed, curvestream is suitable
  - If settings of OSC are needed to reconfigure quickly and frequently, just as mode switch very quickly, curvestream not suitable and curve is better

Scan passive,10GS/s



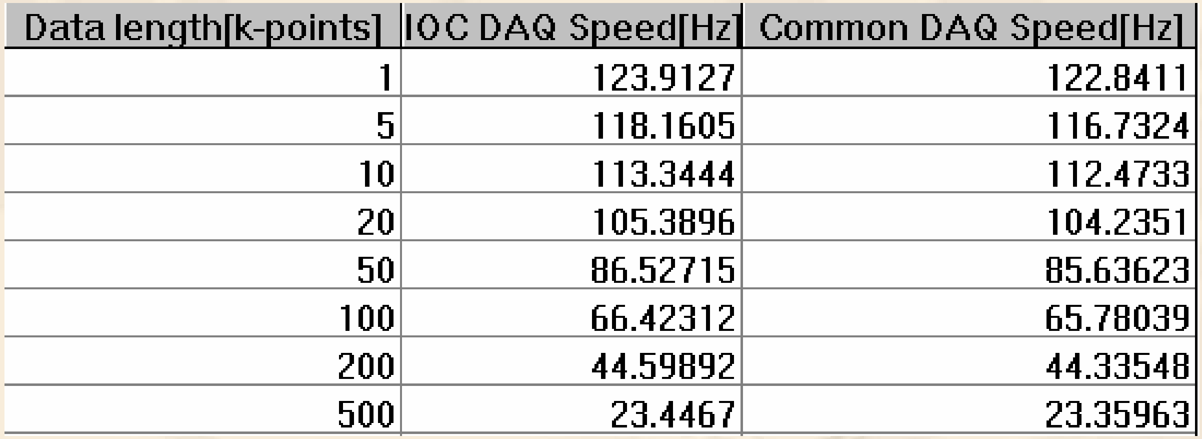
Scan periodic,10GS/s,

- Two cycle
  - Vary scan period (>=0.01s)
  - Vary loop number in read function in device support
- pulse generator
  - 15M,400mv



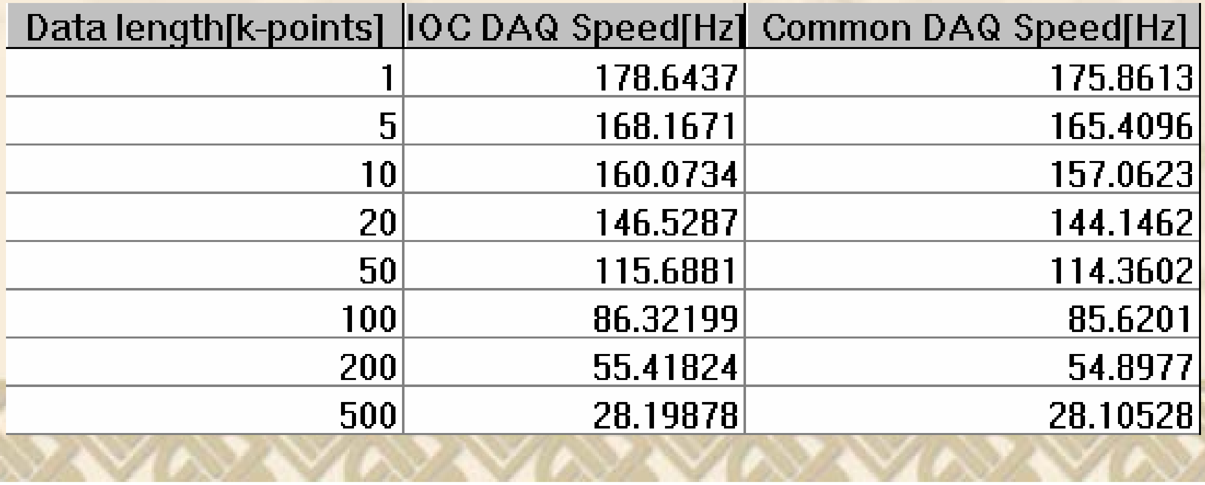
IOC (passive scan) and commom application under same condition

4 channel



Data length[k-points]	IOC DAQ Speed[Hz]	Common DAQ Speed[Hz]
1	123.9127	122.8411
5	118.1605	116.7324
10	113.3444	112.4733
20	105.3896	104.2351
50	86.52715	85.63623
100	66.42312	65.78039
200	44.59892	44.33548
500	23.4467	23.35963

2 channel

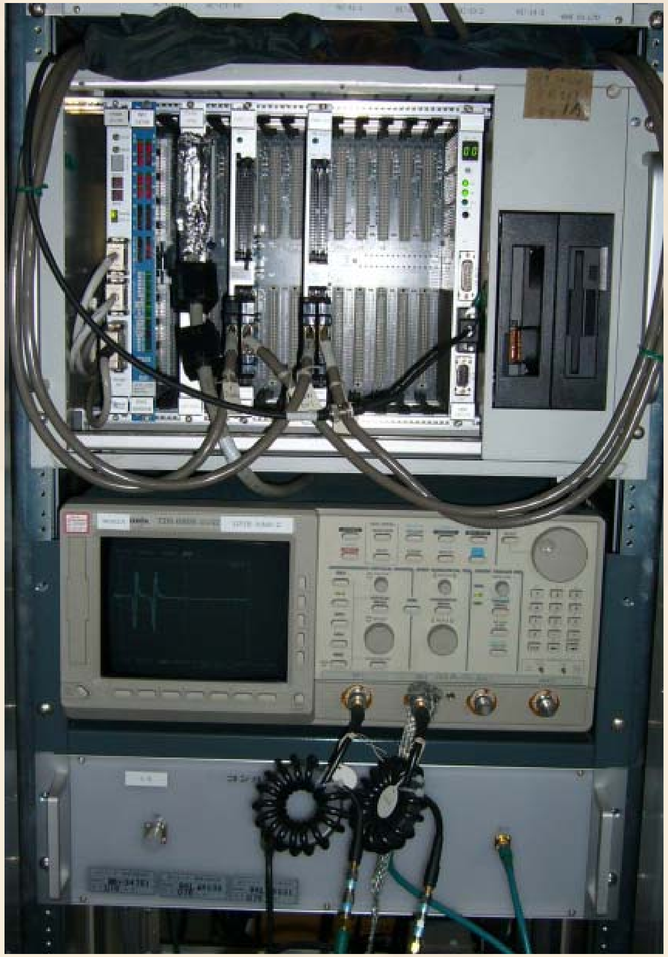


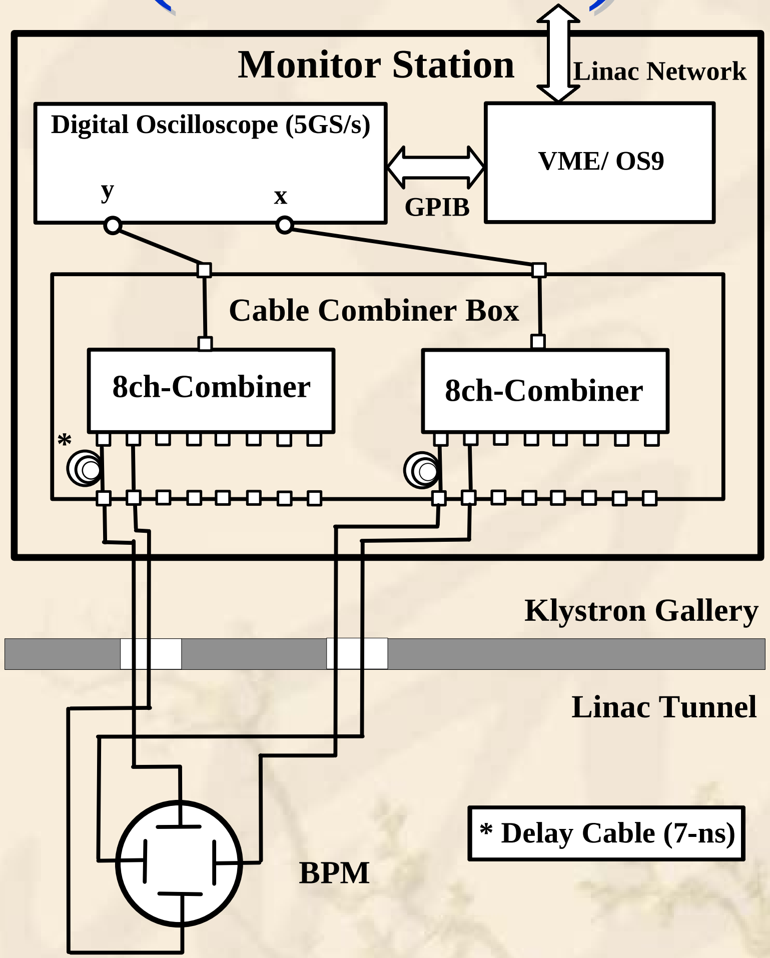
Data length[k-points]	IOC DAQ Speed[Hz]	Common DAQ Speed[Hz]
1	178.6437	175.8613
5	168.1671	165.4096
10	160.0734	157.0623
20	146.5287	144.1462
50	115.6881	114.3602
100	86.32199	85.6201
200	55.41824	54.8977
500	28.19878	28.10528

2. IOC for Upgrading BPM DAQ software

- Now beams of linac are switched to KEKB and PF twice a day
- Next switch frequency is several Hz(1~2Hz)
- Final switch frequency will be 50Hz
- So BPM system and Feedback system will be upgraded to promote stability and quality of beam, speed of switch between KEKB and PF.

Monitor Station (Previous)

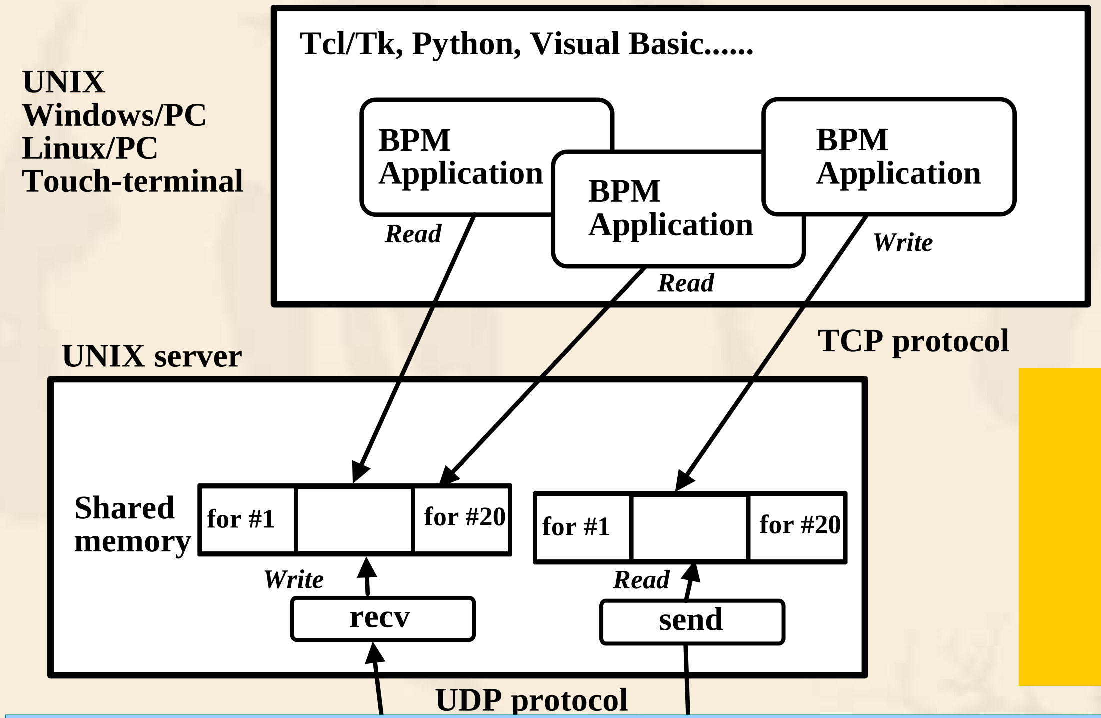




✓VME + Oscilloscope (TectronixTDS680, 5-GSa/s, 8-bits)

✓VME ⇔ Oscilloscope via GPIB (GPIB is slower)

Software structure (Now)



Porting
OS9, Tru64 (lib.)
=> Windows

DAQ Software

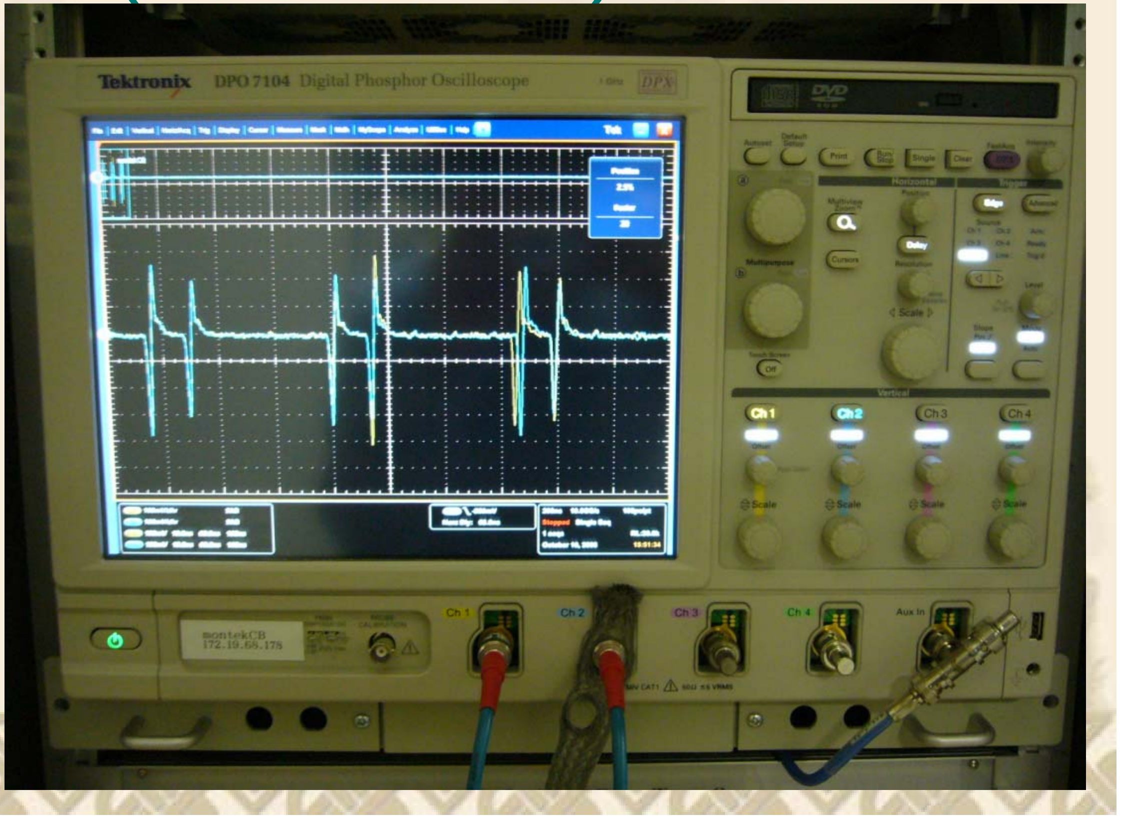
Windows XP | TekVisa

DPO 7104

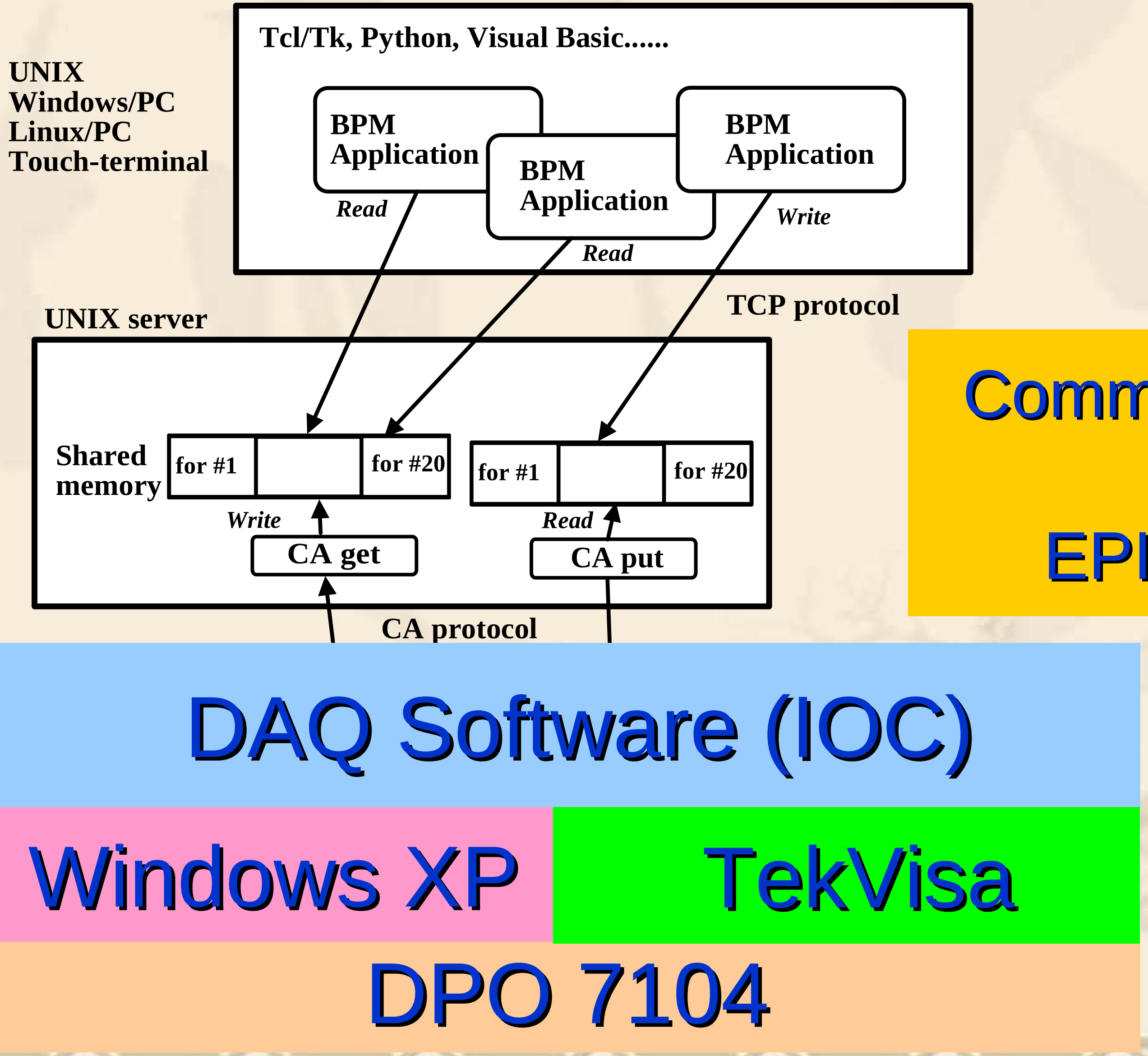
Remove the VME and GPIB, use ethernet

Tektronix DPO 7104

- 10-GSa/s (4ch), 8-bits
- Windows XP based (P4 3.4-GHz)
- Gigabit-Ethernet



EPICS Based



Tcl/Tk, Python, Visual Basic......

UNIX
Windows/PC
Linux/PC
Touch-terminal

BPM Application

BPM Application

BPM Application

Read

Read

Write

TCP protocol

UNIX server

Shared memory

for #1

for #20

for #1

for #20

Write

Read

CA get

CA put

CA protocol

DAQ Software (IOC)

Windows XP

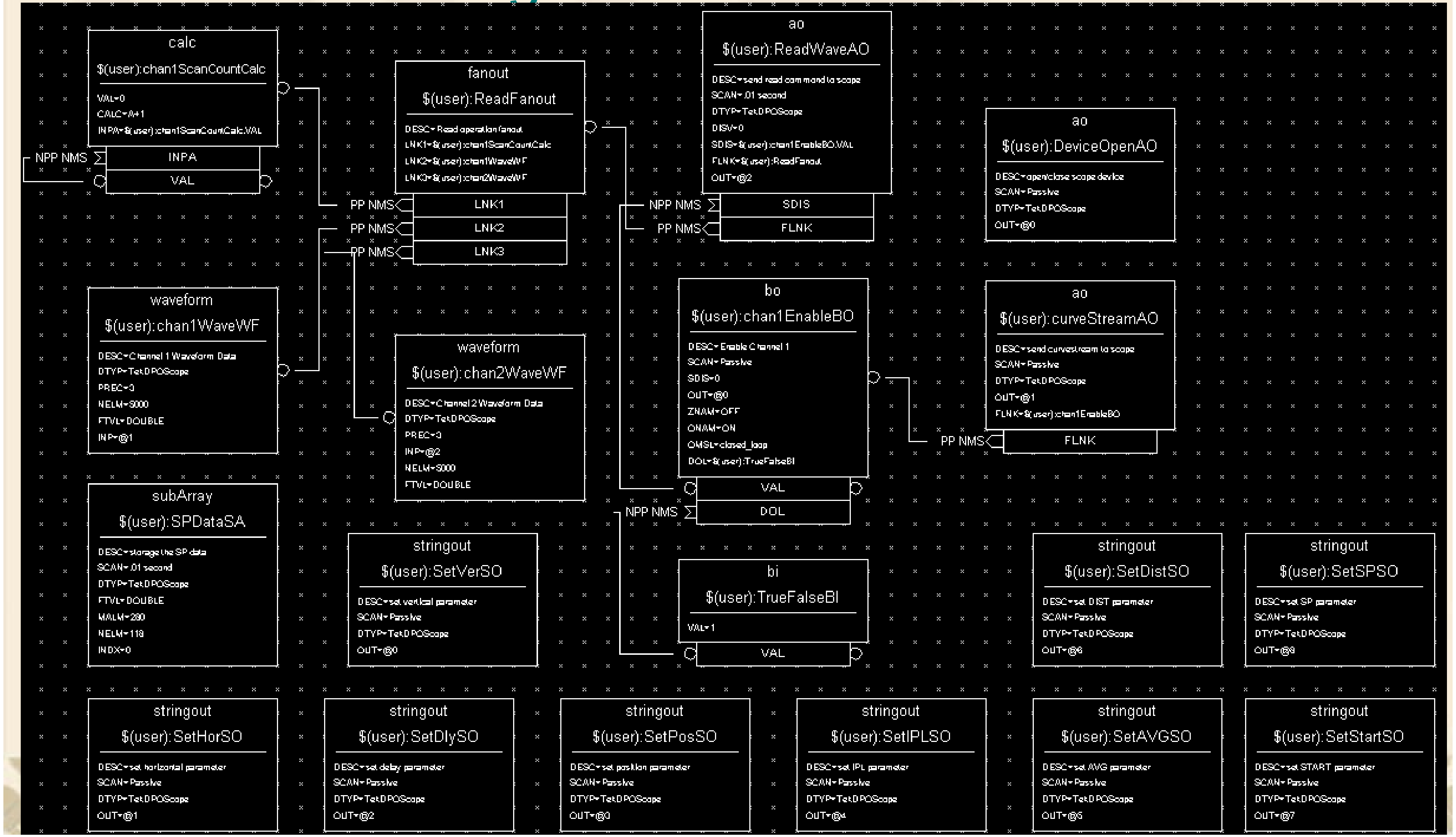
TekVisa

DPO 7104

Common Windows
=>
EPICS based

DAQ Software (IOC)

- VDCT to design the records



calc
$(user):chan1ScanCountCalc
VAL=0
CALC=A+1
INPA=$(user):chan1ScanCountCalc.VAL

fanout
$(user):ReadFanout
DESC=Read operation fanout
LNK1=$(user):chan1ScanCountCalc
LNK2=$(user):chan1WaveWF
LNK3=$(user):chan2WaveWF

ao
$(user):ReadWaveAO
DESC=send read command to scope
SCAN=.01 second
DTYP=TekDPOScope
DISV=0
SDIS=$(user):chan1EnableBO.VAL
FLNK=$(user):ReadFanout
OUT=@2

ao
$(user):DeviceOpenAO
DESC=open/close scope device
SCAN=Passive
DTYP=TekDPOScope
OUT=@0

waveform
$(user):chan1WaveWF
DESC=Channel 1 Waveform Data
DTYP=TekDPOScope
PREC=3
NELM=5000
FTVL=DOUBLE
INP=@1

waveform
$(user):chan2WaveWF
DESC=Channel 2 Waveform Data
DTYP=TekDPOScope
PREC=3
INP=@2
NELM=5000
FTVL=DOUBLE

bo
$(user):chan1EnableBO
DESC=Enable Channel 1
SCAN=Passive
SDIS=0
OUT=@0
ZNAM=OFF
ONAM=ON
OMSL=closed_loop
DOL=$(user):TrueFalseBI

ao
$(user):curveStreamAO
DESC=send curvestream to scope
SCAN=Passive
DTYP=TekDPOScope
OUT=@1
FLNK=$(user):chan1EnableBO

subArray
$(user):SPDataSA
DESC=storage the SP data
SCAN=.01 second
DTYP=TekDPOScope
FTVL=DOUBLE
MALM=280
NELM=118
INDX=0

stringout
$(user):SetVerSO
DESC=set vertical parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@0

bi
$(user):TrueFalseBI
VAL=1

stringout
$(user):SetDistSO
DESC=set DIST parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@6

stringout
$(user):SetSPSO
DESC=set SP parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@8

stringout
$(user):SetHorSO
DESC=set horizontal parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@1

stringout
$(user):SetDlySO
DESC=set delay parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@2

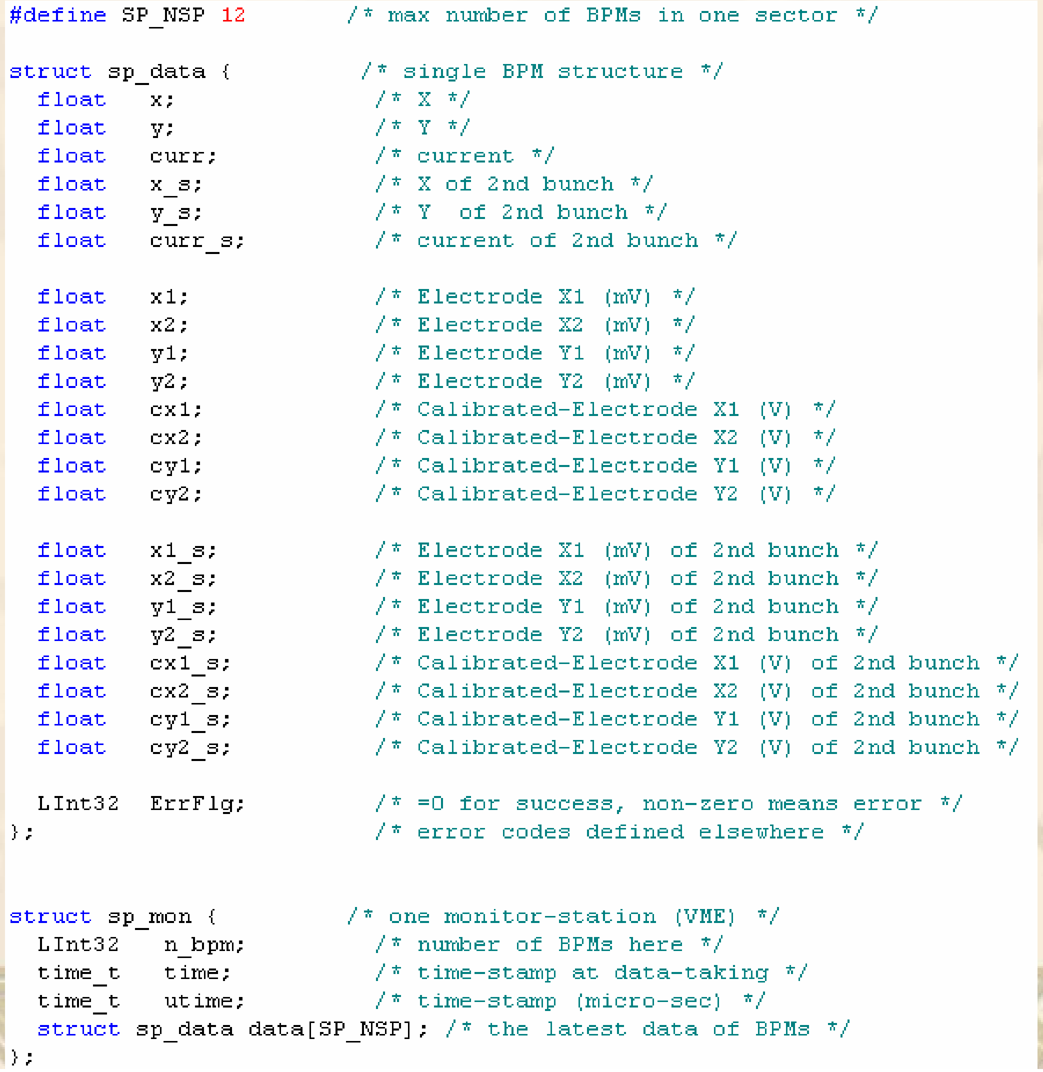
stringout
$(user):SetPosSO
DESC=set position parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@3

stringout
$(user):SetIPLSO
DESC=set IPL parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@4

stringout
$(user):SetAVGSO
DESC=set AVG parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@5

stringout
$(user):SetStartSO
DESC=set START parameter
SCAN=Passive
DTYP=TekDPOScope
OUT=@7

- subArray: store the SP result (position and current value) named $(user):SPDataSA, correspond to struct sp_mon

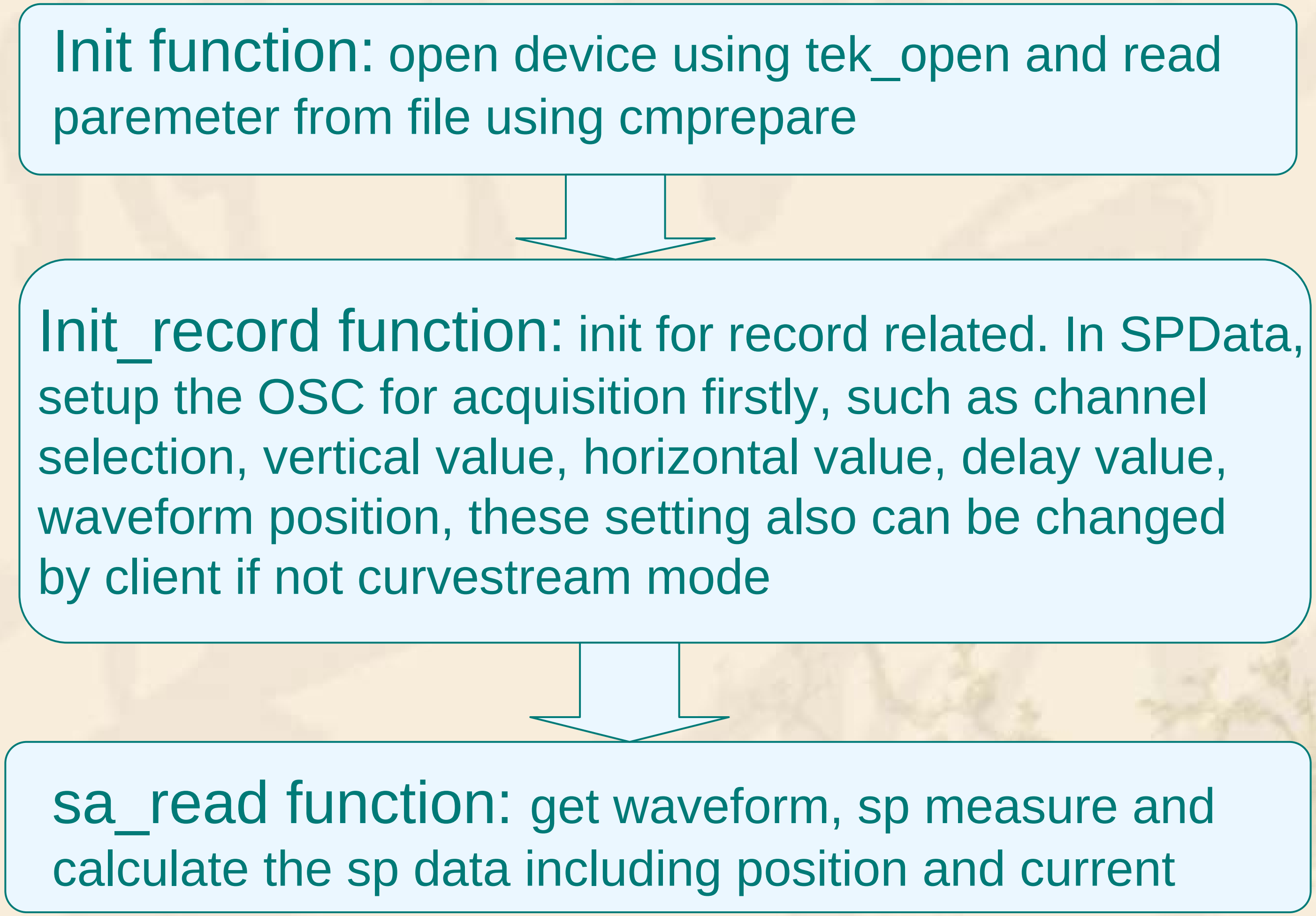


```
#define SP_NSP 12        /* max number of BPMs in one sector */

struct sp_data {          /* single BPM structure */
  float   x;               /* X */
  float   y;               /* Y */
  float   curr;            /* current */
  float   x_s;             /* X of 2nd bunch */
  float   y_s;             /* Y  of 2nd bunch */
  float   curr_s;          /* current of 2nd bunch */

  float   x1;              /* Electrode X1 (mV) */
  float   x2;              /* Electrode X2 (mV) */
  float   y1;              /* Electrode Y1 (mV) */
  float   y2;              /* Electrode Y2 (mV) */
  float   cx1;             /* Calibrated-Electrode X1 (V) */
  float   cx2;             /* Calibrated-Electrode X2 (V) */
  float   cy1;             /* Calibrated-Electrode Y1 (V) */
  float   cy2;             /* Calibrated-Electrode Y2 (V) */

  float   x1_s;            /* Electrode X1 (mV) of 2nd bunch */
  float   x2_s;            /* Electrode X2 (mV) of 2nd bunch */
  float   y1_s;            /* Electrode Y1 (mV) of 2nd bunch */
  float   y2_s;            /* Electrode Y2 (mV) of 2nd bunch */
  float   cx1_s;           /* Calibrated-Electrode X1 (V) of 2nd bunch */
  float   cx2_s;           /* Calibrated-Electrode X2 (V) of 2nd bunch */
  float   cy1_s;           /* Calibrated-Electrode Y1 (V) of 2nd bunch */
  float   cy2_s;           /* Calibrated-Electrode Y2 (V) of 2nd bunch */

  LInt32  ErrFlg;          /* =0 for success, non-zero means error */
};                         /* error codes defined elsewhere */


struct sp_mon {          /* one monitor-station (VME) */
  LInt32   n_bpm;          /* number of BPMs here */
  time_t   time;           /* time-stamp at data-taking */
  time_t   utime;          /* time-stamp (micro-sec) */
  struct sp_data data[SP_NSP]; /* the latest data of BPMs */
};
```

Device support

- Device Type
  - device(waveform,INST_IO,devWfTekDPO,"TekDPOScope")
  - device(ao,INST_IO,devAOTekDPO,"TekDPOScope")
  - device(subArray,INST_IO,devSATekDPO,"TekDPOScope")
  - device(stringout,INST_IO,devSOTekDPO,"TekDPOScope")

subArray device support process



Init function: open device using tek_open and read paremeter from file using cmprepare

Init_record function: init for record related. In SPData, setup the OSC for acquisition firstly, such as channel selection, vertical value, horizontal value, delay value, waveform position, these setting also can be changed by client if not curvestream mode

sa_read function: get waveform, sp measure and calculate the sp data including position and current

CA Client of DAQ IOC software

- OS: Unix, Linux
- Based on the source code of catools of EPICS base which located at base/src/catools
- Based on the application “sprecv” and “shmsem” library
- SP get client
  - Get the SP Data and convert subarray to struct sp_mon and store to share memory when running client once
- SP monitor client
  - Get the SP Data and convert subarray to struct sp_mon and store to share memory when SP Data is changed in IOC
  - Porting “sprecv” to “sp_monitor”

DAQ IOC Software Test

- 50Mhz pulse genrator
- Some setting of OSC is 10Gs/s 100ps/pt ; DIS:WAVE OFF; ACQ:STOPAFTER RUNSTOP; DATA:SOURCE CH1,CH2; :HOR:RECO 20000
- Use curve command
  - Scan periodic: max frequency = 81.54
  - Scan passive: max frequency = 135.14
- Use curvestream command
  - Scan periodic: max frequency = 195.31
  - Scan passive: max frequency = 194.20

Summary

- Different technologies of win32 to develop IOC for different oscilloscope and compare performance of some
  - Develop ioc for waveform acquisition and test performance with the common win32 application
- Develop IOC for upgrading BPM DAQ Software
  - Porting DAQ software to IOC EPICS based and test its performance

Thank you!