The **transferArray** record

H. Kaji (KEK)
Y. Iitsuka, T. Okazaki (East Japan Institute of Technology)

September 2014

1 Introduction for transferArray

The transferArray record is an improvement of the subArray record[1]. We developed this record at KEK for managing the Event Timing System. In this section, the functionality of transferArray is introduced with its structure of fields.

1.1 Functionality

The transferArray record manages elements of waveform record. This record copies arbitrary part of waveform elements and pastes them into another waveform. Figure 1 is the schematic view which explains the functionality of transferArray.

The elements to be copied from the original waveform (**waveform A** in Figure 1) are defined with two variables, "offset" and "size", as subArray does. The offset indicates the first element of the region to be copied and the size indicates number of elements to be copied.

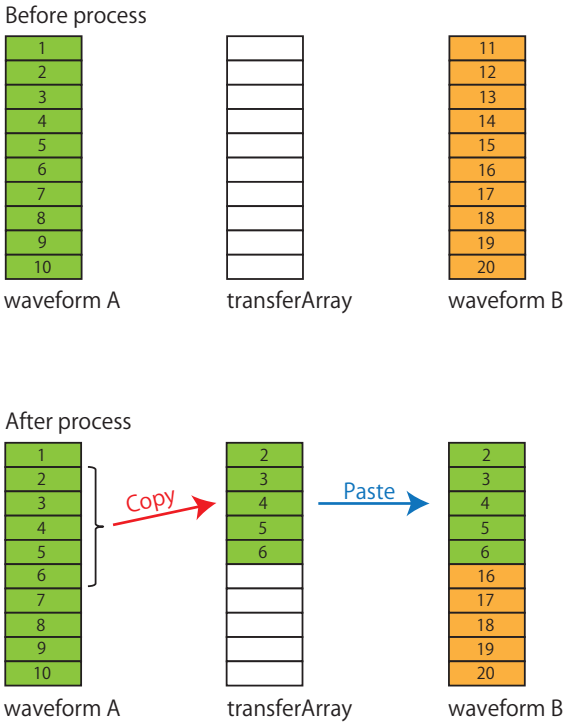

Figure 1: Schematic view of transferArray: this is an example when the transferArray copies elements from **waveform A** and pastes them into **waveform B**. The arbitrary part of original waveform can be copied. The copied elements are stored also into the VAL field of transferArray.

There are some options when transferArray pastes the copied elements into the destination waveform (**waveform B** in Figure 1). They are explained in Figure 2. The copied elements are pasted into the arbitrary region of destination waveform. This region is defined with variable, "offset". There is the flag which define the treatment of the region after pasted elements. The elements of this region are cleared (put "0") where this flag is "ON".

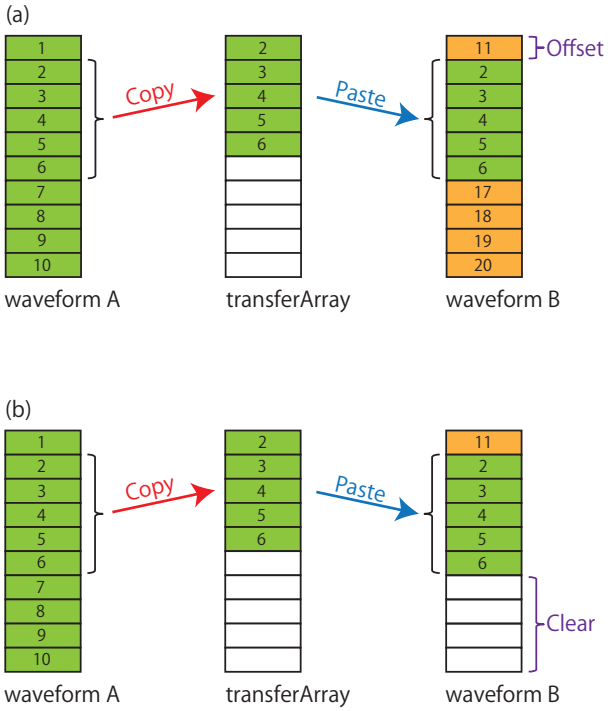

Figure 2: Options for pasting the copied elements: two examples of paste are shown here. In both cases, the copied elements are pasted from the second element of **waveform B**. The arbitrary region of destination waveform can be selected to paste the copied elements with "offset". The elements after pasted region are treated in two ways. The contents of those region are remained in case (a) while they are cleared in case (b). This is optional with the flag.

1.2 Structure of Fields

The structure of transferArray consists of the standard fields of waveform[2] and five special fields. The procedure of copy and paste are described with the following fields:

- INP:

  The name of original waveform is specified here. The role of this field is same as that of subArray.

- OUT: **for transferArray**
  The name of destination waveform is specified here.

- NELM:
  This field defines the size of transferArray and consequently limits the size of copied elements.

- TASI: **for transferArray**
  Offset for original waveform ($0 \sim N$). This field defines from which element of original waveform are copied. The transferArray record copies from the first element of original waveform where "0" is filled. The role of this field is same as INDX of subArray.

- TATC: **for transferArray**
  Number of elements to be copied ($1 \sim N$). This field defines how many elements are copied. The role of this field is same as NELM of subArray.

- TADI: **for transferArray**
  Offset for destination waveform ($0 \sim N$). This field defines from which element of destination waveform the copied elements are stored. The transferArray record stored copied elements from the first element of destination waveform where "0" is filled.

- TAZF: **for transferArray**
  Flag for clear (0 or 1). At the destination waveform, the elements after the pasted region are clear where "1" is filled.

where "OUT", "TASI", "TADI", "TATC", and "TAZF" are special fields for transferArray. The further specifications for these field are summarized in Table 1.

Field	Type	DCT	Initial	Access	Modify	Rec Proc Monitor	PP
OUT	OUTLINK	Yes	0	No	No	N/A	No
TASI	ULONG	Yes	0	Yes	No	No	No
TADI	ULONG	Yes	0	Yes	No	No	No
TATC	ULONG	Yes	0	Yes	No	No	No
TAZF	ULONG	Yes	0	Yes	No	No	No

Table 1: Detailed specifications for special field of transferArray

1.3 Behavior in the anomalous setting

Here we summarize the behavior of transferArray when its fields is set anomalously. The following five conditions are anomalous setting of transferArray.

- when “TASI” is larger than “NELM” of original waveform,
  Error! transferArray is not processed.
- “TASI+TATC” is larger than “NELM” of original waveform,
  only “NELM−TASI” elements are copied. See Figure 3 (a).
- “TADI” is larger than “NELM” of destination waveform,
  Error! transferArray is not processed.
- “TADI+TATC” is larger than “NELM” of destination waveform,
  only “NELM−TADI” elements are pasted. See Figure 3 (b).
- “TATC” is larger than “NELM” of transferArray,
  Error! transferArray is not processed.

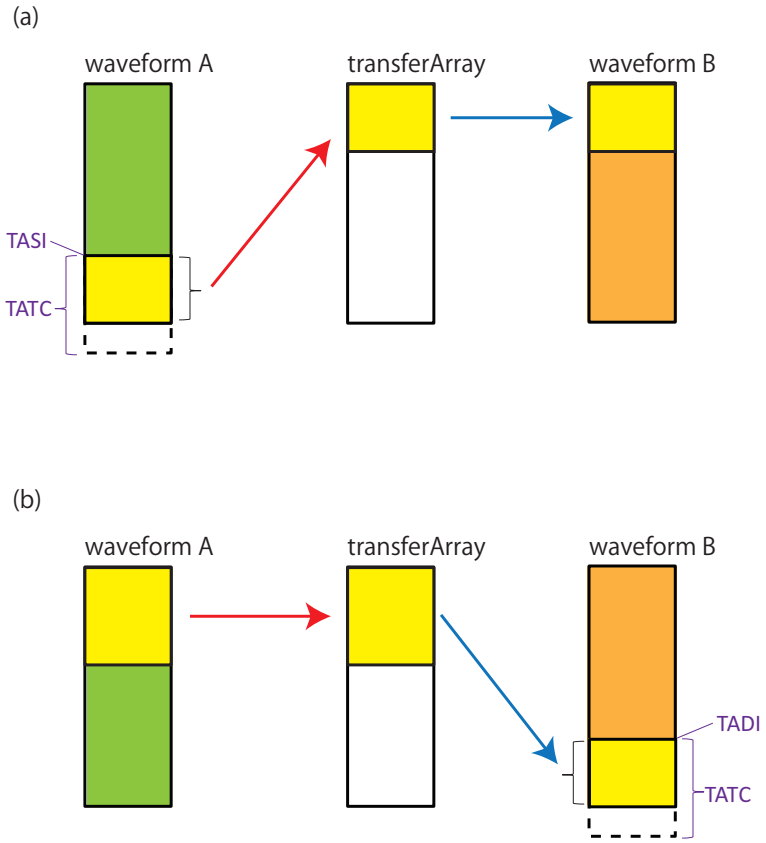

Figure 3: Behavior in the anomalous setting of transferArray: the behavior where “TASI+TATC”>“NELM(**waveform A**)” is shown in (a) and that where “TADI+TATC”>“NELM(**waveform B**)” is shown in (b), respectively.

2 Installation

The transferArray record is available only by installing the new EPICS module which we developed at KEK. We explain the procedure of installation in this section.

2.1 Download

The module can be download on the following web site.

http:www-linac.kek.jp/cont/epics/transferarray/

It includes the **src** and **Db** directories in the **transferArrayApp** directory. All sources are stored in the **src** directory with the sample template of Makefile. An example of db file is available in the **Db** directory. The tar ball, **transferArray.tar.gz**, which includes all above components also can be downloaded on this site.

2.2 Procedure for Installation

The compile and link can be implemented by adding small descriptions on your Makefile to build EPICS IOC.

Add following description on your Makefile:
DBDINC += transferArrayRecord

Add following description at *yourIoc*_DBD:
*yourIoc*_DBD += transferArrayRecord.dbd

Add following description at *yourIoc*_SRCS:
*yourIoc*_SRCS += transferArrayRecord.c

Then, compile at your <TOP> directory.

3 Examples for Event Timing System

Here we introduced a practical example of transferArray for managing the sequence of Event Generator (EVG)[3]. The stability and reliability of these examples for the EVG operation has already confirmed with the condition shown in Table 2.

The setup of sequence is implemented with the waveform record in the MRFIOC2 device/driver[4]. Up to 2047 Events can be scheduled with their delivering timing on

Items		Version
CPU	Hardware	MVME5500
	OS	VxWorks 6.8.3
EPICS	Base	3.14.12.1
	Device/driver	MRFIOC2[4]
EVG	Hardware	VME-EVG-230[3]
	Firmware	0005
EVR	Hardware	VME-EVR-230RF[3]
	Firmware	0005

Table 2: Versions for software and hardware where the stability and reliability of example IOC records are confirmed.

one sequence of EVG. The transferArray record is utilized to change the contents of waveform so that the setup of sequence is changed.

An example set of basic records to configure the EVG is shown in Appendix A and Appendix B is the set of records for changing the EVG configuration. It works when one of following records are processed.

```
$(USER):PPM:FANOUT1
$(USER):PPM:FANOUT2
$(USER):PPM:FANOUT3
$(USER):PPM:FANOUT4
```

The four kinds of sequences are prepared in the waveform records "in advance" and one of them are selected and set into EVG with the processes of transferArray records.

For example, KEK LINAC provides four different kinds of beams into four experimental rings and it switches beam modes every injection with 50 Hz. It is realized by switching the EVG configuration in the every injection pulse.

References

[1] subArray web page:
https://www.slac.stanford.edu/grp/ssrl/spear/epics/base/RecordRef/Recordref-39.html

[2] waveform web page:
https://www.slac.stanford.edu/grp/ssrl/spear/epics/base/RecordRef/Recordref-43.html

[3] MRF VME Event Timing Module web page:
http://www.mrf.fi/index.php/vme-products

[4] MRFIOC2 web page:
http://www.sourceforge.net/mrfioc2/

Appendix A

Basic Record to Control EVG

```
record(bo, "$(USER):SoftSeq0:Enable-Cmd") {
  field(DTYP, "EVG ENABLE SEQ")
  field(DESC, "Enable EVG Sequence")
  field(OUT, "#C S0 @$(Module_ID)")
  field(VAL, "1")
  field(UDF, "0")
  field(ZNAM, "0")
  field(ONAM, "1")
}
record(bo, "$(USER):SoftSeq0:Disable-Cmd") {
  field(DTYP, "EVG DISABLE SEQ")
  field(DESC, "Disable EVG Sequence")
  field(OUT, "#C S0 @$(Module_ID)")
  field(VAL, "0")
  field(UDF, "0")
  field(ZNAM, "0")
  field(ONAM, "1")
}
record(mbbo, "$(USER):SoftSeq0:RunMode-Sel") {
  field(DTYP, "EVG SEQ RUN MODE")
  field(OUT, "#C S0 @$(Module_ID)")
  field(UDF, "0")
  field(ZRST, "Normal")
  field(ONST, "Automatic")
  field(TWST, "Single")
  field(THSV, "INVALID")
  field(FRSV, "INVALID")
  field(FVSV, "INVALID")
  field(SXSV, "INVALID")
  field(SVSV, "INVALID")
  field(EISV, "INVALID")
  field(NISV, "INVALID")
  field(TESV, "INVALID")
  field(ELSV, "INVALID")
  field(TVSV, "INVALID")
  field(TTSV, "INVALID")
  field(FTSV, "INVALID")
  field(FFSV, "INVALID")
  field(UNSV, "INVALID")
```

```
  field(IVOA, "Don't drive outputs")
}
record(bo, "$(USER):SoftSeq0:TsInpMode-Sel") {
  field(DTYP, "EVG SEQ INPUT MODE")
  field(DESC, "EVG Sequence")
  field(OUT, "#C S0 @$(Module_ID)")
  field(VAL, "1")
  field(UDF, "0")
  field(ZNAM, "EGU")
  field(ONAM, "TICKS")
}
record(mbbo, "$(USER):SoftSeq0:TsResolution-Sel") {
  field(DTYP, "EVG SEQ TS RES")
  field(OUT, "#C S0 @$(Module_ID)")
  field(ZRST, "Sec")
  field(ONST, "mSec")
  field(TWST, "uSec")
  field(THST, "nSec")
  field(ZRVL, "0")
  field(ONVL, "3")
  field(TWVL, "6")
  field(THVL, "9")
  field(FRSV, "INVALID")
  field(FVSV, "INVALID")
  field(SXSV, "INVALID")
  field(SVSV, "INVALID")
  field(EISV, "INVALID")
  field(NISV, "INVALID")
  field(TESV, "INVALID")
  field(ELSV, "INVALID")
  field(TVSV, "INVALID")
  field(TTSV, "INVALID")
  field(FTSV, "INVALID")
  field(FFSV, "INVALID")
  field(UNSV, "INVALID")
  field(IVOA, "Don't drive outputs")
}
record(bo, "$(USER):SoftSeq0:Unload-Cmd") {
  field(DTYP, "EVG UNLOAD SEQ")
  field(DESC, "Dealloc EVG Sequence")
  field(OUT,  "#C S0 @$(Module_ID)")
  field(VAL,  "1")
```

```
  field(UDF,  "0")
  field(ZNAM, "0")
  field(ONAM, "1")
}
record(bo, "$(USER):SoftSeq0:Load-Cmd") {
  field(DTYP, "EVG LOAD SEQ")
  field(DESC, "ALLOC EVG Sequence")
  field(OUT,  "#C S0 @$(Module_ID)")
  field(VAL,  "1")
  field(UDF,  "0")
  field(ZNAM, "0")
  field(ONAM, "1")
}
record(bo, "$(USER):SoftSeq0:Commit-Cmd") {
  field(DTYP, "EVG COMMIT SEQ")
  field(DESC, "Commit EVG Sequence")
  field(OUT,  "#C S0 @$(Module_ID)")
  field(VAL,  "1")
  field(UDF,  "0")
  field(ZNAM, "0")
  field(ONAM, "1")
}
record(mbbo, "$(USER):SoftSeq0:TrigSrc-Sel") {
  field(DTYP, "EVG SEQ TRIG SRC")
  field(OUT, "#C S0 @$(Module_ID)")
  field(UDF, "0")
  field(ZRST, "None")
  field(ONST, "Mxc0")
  field(TWST, "Mxc1")
  field(THST, "Mxc2")
  field(FRST, "Mxc3")
  field(FVST, "Mxc4")
  field(SXST, "Mxc5")
  field(SVST, "Mxc6")
  field(EIST, "Mxc7")
  field(NIST, "AC")
  field(TEST, "Software")
  field(ELST, "Front0")
  field(TVST, "Front1")
  field(TTST, "Univ0")
  field(FTST, "Univ1")
  field(FFST, "Univ2")
```

```
  field(ZRVL, "31")
  field(ONVL, "0")
  field(TWVL, "1")
  field(THVL, "2")
  field(FRVL, "3")
  field(FVVL, "4")
  field(SXVL, "5")
  field(SVVL, "6")
  field(EIVL, "7")
  field(NIVL, "16")
  field(TEVL, "19")
  field(ELVL, "41")
  field(TVVL, "45")
  field(TTVL, "42")
  field(FTVL, "46")
  field(FFVL, "50")
  field(PINI, "YES")
  field(VAL, "9")
}
record(waveform, "$(USER):SoftSeq0:EvtCode-SP") {
  field(DTYP, "EVG SEQ EVENT CODE")
  field(DESC, "Sequence event code array")
  field(INP, "#C S0 @$(Module_ID)")
  field(NELM, "2047")
  field(FTVL, "UCHAR")
}
record(waveform, "$(USER):SoftSeq0:Timestamp-SP") {
  field(DTYP, "EVG SEQ TIMESTAMP")
  field(DESC, "Sequence timestamp array")
  field(INP, "#C S0 @$(Module_ID)")
  field(NELM, "2047")
  field(FTVL, "DOUBLE")
}
```

Appendix B

An example for Pulse-to-Pulse Modulation

```
record(fanout, "$(USER):PPM:FANOUT1"){
  field(SELM, "All")
  field(LNK1, "$(USER):TA:BM1:EvList")
  field(LNK2, "$(USER):TA:BM1:TimeList")
  field(LNK3, "$(USER):PPM:FANOUT_SET"){
}
record(fanout, "$(USER):PPM:FANOUT2"){
  field(SELM, "All")
  field(LNK1, "$(USER):TA:BM2:EvList")
  field(LNK2, "$(USER):TA:BM2:TimeList")
  field(LNK3, "$(USER):PPM:FANOUT_SET"){
}
record(fanout, "$(USER):PPM:FANOUT3"){
  field(SELM, "All")
  field(LNK1, "$(USER):TA:BM3:EvList")
  field(LNK2, "$(USER):TA:BM3:TimeList")
  field(LNK3, "$(USER):PPM:FANOUT_SET"){
}
record(fanout, "$(USER):PPM:FANOUT4"){
  field(SELM, "All")
  field(LNK1, "$(USER):TA:BM4:EvList")
  field(LNK2, "$(USER):TA:BM4:TimeList")
  field(LNK3, "$(USER):PPM:FANOUT_SET"){
}
record(fanout, "$(USER):PPM:FANOUT_SET"){
  field(SELM, "All")
  field(LNK1, "$(USER):SoftSeq0:Unload-Cmd")
  field(LNK2, "$(USER):SoftSeq0:Load-Cmd")
  field(LNK3, "$(USER):SoftSeq0:Commit-Cmd")
}


record(transferArray, "$(USER):TA:BM1:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
  field(INP,  "$(USER):BM1:EvList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:EvtCode-SP PP")
```

```
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM1:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
  field(INP,  "$(USER):BM1:TimeList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:Timestamp-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM2:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
  field(INP,  "$(USER):BM2:EvList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:EvtCode-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM2:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
  field(INP,  "$(USER):BM2:TimeList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:Timestamp-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM3:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
  field(INP,  "$(USER):BM3:EvList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:EvtCode-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
```

```
}
record(transferArray, "$(USER):TA:BM3:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
  field(INP,  "$(USER):BM3:TimeList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:Timestamp-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM4:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
  field(INP,  "$(USER):BM4:EvList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:EvtCode-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}
record(transferArray, "$(USER):TA:BM4:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
  field(INP,  "$(USER):BM4:TimeList")
  field(TASI, "0")
  field(TATC, "10")
  field(OUT,  "$(USER):SoftSeq0:Timestamp-SP PP")
  field(TADI, "0")
  field(TAZF, "1")
}


record(waveform, "$(USER):BM1:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
}
record(waveform, "$(USER):BM1:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
}
record(waveform, "$(USER):BM2:EvList"){
```

```
  field(NELM, "10")
  field(FTVL, "UCHAR")
}
record(waveform, "$(USER):BM2:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
}
record(waveform, "$(USER):BM3:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
}
record(waveform, "$(USER):BM3:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
}
record(waveform, "$(USER):BM4:EvList"){
  field(NELM, "10")
  field(FTVL, "UCHAR")
}
record(waveform, "$(USER):BM4:TimeList"){
  field(NELM, "10")
  field(FTVL, "DOUBLE")
}
```