Chapter 1

Dish for babies

1.1 babies modes

There are four modes for babies.

- l normal Linux mode
- r RTLinux mode
- c Shared memory mode
- s Scaler mode
- d Dummy driver mode

For normal Linux and RTLinux modes, babies receives data from device drivers. Shared memory, Scaler and Dummy driver modes communicate with other process (user program) via shared memory. Figure 1.1 shows the way to communicate with babies. Scaler and dummy driver modes are omitted. Their methods are similar to shared memory mode one.

Figure 1.1: Data and command communication method for babies.
1.2 Shared memory mode

To communicate with other user processes, two regions of shared memory and one FIFO are used.

The following is a list of important variables in babies.

For Data (shmpt)
- Shared memory Key = EFshmKEY + EFN = 523400 + EFN
- Size = EF_SHM_SIZE = 524300

For Handshake (fifofd)
- FIFO path = COMFIFO = /tmp/babirldbfifo

EFshmKEY, EF_SHM_SIZE, and EFSEMKEY are written in include/bi-config.h.

This FIFO will be created by babies automatically. shmpt, and fifofd are memory pointer and file handler in babies.c, respectively.

1.2.1 Start sequence

At the start time, babies changes shmpt + EF_SHM_RUN to be ‘1’. (In the source code of babies, shmpt + EF_SHM_RUN is defined as *babiesrun.) Next, user program will do its initialization routine. When this initialization is finished, shmpt + EF_SHM_SSF have to be ‘1’ by user program. babies do not send ‘start ack’ to babild until shmpt + EF_SHM_SSF is changed to be ‘1’.

Before start
- shmpt + EF_SHM_RUN = 0
- shmpt + EF_SHM_SSF = 0

Start time
- shmpt + EF_SHM_RUN = 1 (by babies)
- shmpt + EF_SHM_SSF = 1 (by user program)

1.2.2 Data copy sequence

To serve data to babies from user program, handshake with FIFO is used. shmpt + EF_SHM_FLAG1 and shmpt + EF_SHM_FLAG2 are flags for data occupancy of buffer. When both flags are ‘1’ which means “both buffer is full”, user program must not copy data to shared memory. If shmpt + EF_SHM_FLAG1 and/or shmpt + EF_SHM_FLAG2 are ‘0’, user program can copy data to shared memory. Memory pointers are shmpt + EF_SHM_DATA1 for buffer 1 and shmpt + EF_SHM_DATA2 for buffer 2. When data is copied to buffer 1, user program also puts ‘0’ to the FIFO. For buffer 2 case, user program have to put ‘1’ to the FIFO. babies receives the values of ‘0’ or ‘1’ from the FIFO. Next, babies send data to babild from shmpt + EF_SHM_DATA1/2. After the data transfer, babies writes ‘0’ to shmpt + EF_SHM_FLAG1/2.

User program
- check shmpt + EF_SHM_FLAG1/2
- if 0, copy data to shmpt + EF_SHM_DATA1/2
- put 0 or 1 to FIFO

babies
- polling FIFO
- when the values of 0 or 1 come, send data to babild from shmpt + EF_SHM_DATA1/2
- write 0 to shmpt + EF_SHM_FLAG1/2

1.2.3 Stop sequence

When babies receives the stop command from babild, *babiesrun is changed to be ‘0’. And then, user program launches stop routine. At the end of stop routine, user program writes shmpt + EF_SHM_SSF = 0 and puts ‘-1’ to the FIFO.
1.2. SHARED MEMORY MODE

Before stop (running)

shmptr + EF_SHM_RUN = 1
shmptr + EF_SHM_SSF = 1

Stop time

shmptr + EF_SHM_RUN = 0 (by babies)
user program do stop routine
shmptr + EF_SHM_SSF = 0 (by user program)
put -1 to FIFO (by user program)

1.2.4 dexecuter

‘devtool/dexecuter.c’ is a good example for this shared memory mode. This program generates dummy data with given event rate and size. You can test this by following commands:

1. babiau/babiau
2. babies/babies -c 10
3. devtool/dexecuter 10 100 100 30 5000
4. babicon/babicon

In this case, the event fragment number is 10. Arguments of dexecuter are listed:

dexecuter EFN EVTRATE EVTLEN DISPERSION EFSIZE
EFN : Event Fragment Number 0-254
EVTRATE : Event rate 1-10000 (evt/s)
EVTLEN : Length of one event 1-200 (short word)
DISPERSION : Dispersion of the length of one event 1-100 (short word)
EFSIZE : Block size 1-20000 (short word)

To terminate dexecuter, please enter Ctrl-C.

‘devtool/dexecuterts.c’ is almost the same as dexecuter, except for its event header includes time stamp. Within dexecuterts, the time stamp value is equal to the event number.