CyFlex® Knowledge Article

Floger Data Collection with Pre and Post Trigger

Author: Leonard Logterman

January 15, 2014

The data logger task `floger` can be used to record data prior to a specified ‘trigger’ event. It can also record data after the ‘trigger’ event. The ‘stop event’ or the ‘release event’ specified in the `floger` spec file is used as the ‘trigger’ event Either or both events may be specified. This capability is similar to a storage oscilloscope that has pre-trigger and post-trigger capability.

Three keywords are associated with this capability. In addition, several other keywords must be specified before this capability is functional. The keywords associated with this capability are:

```
@FIFO_LOG_BUFFER
@FIFO_POST_TRIGGER_INTERVAL
    <interval>
@FIFO_POST_TRIGGER_SCANS
    <number of scans>
```

The `@FIFO_LOG_BUFFER` keyword is required, while the other two are optional. In addition, the `@MAX_SCANS` keyword and either the `@STOP_EVENT` or `@RELEASE_EVENT` must be specified or both may be specified.

Pre-Trigger Data Collection

Data collection begins when the `@START_EVENT` is received and the values are stored in memory. If no `@START_EVENT` is specified, data collection begins when the `floger` task is spawned. Data is collected until `@MAX_SCANS` have been taken. After that, the oldest scan is thrown away and the new scan is added to the data. In other words, the data is being stored in a First In First Out (FIFO) buffer in memory. The size of the memory buffer, in terms of time, is a function of `@MAX_SCANS` and the `@SCAN_INTERVAL`. The process of throwing away the oldest scan and adding the newest scan continues until a ‘trigger’ event is received. The ‘trigger’ event is either the `@STOP_EVENT` or the `@RELEASE_EVENT`.

When a ‘trigger’ event is received, the buffer is written to the hard disk using the file name that was specified. The action taken by `floger` after the data is written to the file depends on which ‘trigger’ was received and whether or not a `@START_EVENT` was specified. If the ‘trigger’ event was the `@RELEASE_EVENT`, `floger` exits. However, if the ‘trigger’ event was the `@STOP_EVENT`, the action depends on whether or not a `@START_EVENT` was specified. If a `@START_EVENT` was specified, `floger` will remain idle until another `@START_EVENT` is received. Otherwise, `floger` will continue collecting data and storing it in the buffer In both cases, when data collection starts, it is added to the buffer in the FIFO fashion.

The normal action of `floger` is to exit when no `@START_EVENT` is specified after the data file is written. However, this is not the case when `floger` is collecting data in the FIFO mode. As a result, the `@RELEASE_EVENT` should be specified in order to have a clean method of removing `floger` from the system.

Post-Trigger Data Collection

A ‘post’ trigger section can be included in the data buffer. The post trigger section is specified when either the `@FIFO_POST_TRIGGER_INTERVAL` keyword or the `@FIFO_POST_TRIGGER_SCANS` keyword is entered. If the ‘interval’ keyword is specified, a time value consistent with the specified maximum scans and scan interval should be specified. If the ‘scans’ keyword is specified, a value smaller than the specified maximum scans should be entered.

The size of the data buffer is defined by the `@MAX_SCANS` keyword. Therefore, when post trigger operation is specified, the number of pre-trigger scans in the buffer is reduced. The actions performed by `floger` as a function of an event for the various conditions described above is shown in the table below. The alphabetic characters in parenthesis under the ‘Start Event Specified’ column correspond to the alphabetic characters used in the time line discussed in section **Data Collection Time Line** .

Initiating Event	Start Event Specified	Stop Event Specified	Release Event Specified	Post Trigger Specified	Action Performed
`floger` is Spawned	No	--	--	--	1. Data Collection Begins
	Yes (a)	--	--	--	1. Waits for Start Event
Start Event Received	Yes (b)	--	--	--	1. Data Collection Begins
Stop Event Received	No	Yes	--	No	1. File Written 2. Data Collection Continues
	Yes	Yes	--	No	1. Write File Wait for Start Event
	No	Yes	--	Yes	1. Continue Data Collection for ‘Post Trigger’ scans 2. Write File 3. Data Collection Continues
	Yes (d)	Yes	--	Yes	1. Continue Data Collection for ‘Post Trigger’ scans 2. File written 3. Wait for Start event
Release Event Received	--	--	Yes	No	1. File Written 2. Task Exits
	-- (g)	--	Yes	Yes	1. Continue Data Collection for ‘Post Trigger’ scans 2. Write File 3. Task Exits

Data Collection Timeline

The following depicts a timeline of data collection when the following keywords are included in the spec file. Each dash (-) indicates a ‘scan’ of all the specified channels.

```
@FIFO_LOG_BUFFER
@FIFO_POST_TRIGGER_SCANS
     5
@MAX_SCANS
    10
@START_EVENT
    fifo_strt
@STOP_EVENT
   fifo_stop
@RELEASE_EVENT
   fifo_rels
@SCAN_INTERVAL
   .5[sec]


-----------------------------
^ ^       ^ ^    ^ ^  ^    ^
a b       c d    e f  g    h
```

point a

The `floger` task is spawned and the spec file is read. It waits to receive the `start_event` to start collecting data.

point b

The event `fifo_strt` is received and the buffer begins to be filled.

point c

The maximum number of scans has been reached. The additional scans will be added and the oldest scan will be discarded.

point d

The event `fifo_stop` is received. Since a `@FIFO_POST_TRIGGER_SCANS` of five is specified, collection of data continues for five more scans.

point e

Five post trigger scans have been collected and the buffer is written to disk. The file contains five scans that were collected before the ‘trigger’ event and five scans that were collected after the ‘trigger’ event. Since a start event was specified, data collection stops until another start event is received.

point f

Another start event is received and data collection resumes.

point g

The event `fifo_rels` is received. Since there `@FIFO_POST_TRIGGER_SCANS` of five is specified, collection of data continues for five more scans.

point h

Five post trigger scans have been collected and the buffer is written to disk. Looking at the time line, the `fifo_rels` event was received four scans after the start event was received. As a result, the first scan in the file written to disk will be the same as the last scan in the first file. Since the received event was the release event, the task exits.