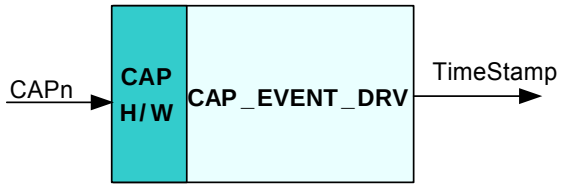
CAP_EVENT_DRV	*Capture Input Event Driver*

Description

This module provides the instantaneous value of the selected time base (GP Timer) captured on the occurrence of an event. Such events can be any specified transition of a signal applied at the event manager (EV) capture input pins of 281x devices or ECAP input pins of 280x devices.



Availability

This 16-bit module is available in one interface format:

1) The C interface version

Module Properties

Type: Target Dependent, Application Independent

Target Devices: x281x or x280x

C Version File Names: f281xcap.c, f281xcap.h (for x281x)
f280xcap.c, f280xcap.h (for x280x)

IQmath library files for C: N/A

Item	C version	Comments
Code Size□ (x281x/x280x)	32/34 words	
Data RAM	0 words•	
xDAIS ready	No	
XDAIS component	No	IALG layer not implemented
Multiple instances	Yes	
Reentrancy	Yes	

• Each pre-initialized CAPTURE structure consumes 6 words in the data memory

□ Code size mentioned here is the size of the ***init()*** and ***read()*** functions

C Interface

Object Definition

The structure of CAPTURE object is defined by following structure definition for

x281x series

```
typedef struct { Uint32 TimeStamp;      // Output: Timer value when capture is detected (Q0)
                 void (*init)();        // Pointer to the init function
                 Uint16 (*read)();      // Pointer to the read function
               } CAPTURE;
```

x280x series

```
typedef struct { int32 EventPeriod;     // Output: Timer value difference between two edges (Q0)
                 void (*init)();        // Pointer to the init function
                 Uint16 (*read)();      // Pointer to the read function
               } CAPTURE;
```

```
typedef CAPTURE *CAPTURE_handle;
```

Item	Name	Description	Format	Range(Hex)
Inputs	CAPn (n=1,2,3,4)	Capture input signals to 281x device	N/A	0-3.3 v
Outputs	TimeStamp (x281x)	TimeStamp for capture unit FIFO registers.	0	00000000-0000FFFF
	EventPeriod (x280x)	Timer value difference between two edges detected.	0	80000000-7FFFFFFF

Special Constants and Data types

CAPTURE
The module definition is created as a data type. This makes it convenient to instance an interface to the CAPTURE driver. To create multiple instances of the module simply declare variables of type CAPTURE.

CAPTURE_handle
User defined Data type of pointer to CAPTURE module

CAPTURE_DEFAULTS
Structure symbolic constant to initialize CAPTURE module. This provides the initial values to the terminal variables as well as method pointers.

Methods

void F281X_EV1_CAP_Init(CAPTURE *);
void F281X_EV1_CAP_Read(CAPTURE *);

void F280X_CAP_Init(CAPTURE *);
void F280X_CAP_Read(CAPTURE *);

This default definition of the object implements two methods – the initialization and the runtime compute function for CAPTURE generation. This is implemented by means of a function pointer, and the initializer sets this to F281X_EV1_CAP_Init and F281X_EV1_CAP_Read functions for x281x or F280X_CAP_Init and F280X_CAP_Read functions for x280x. The argument to this function is the address of the CAPTURE object.

Module Usage

Instantiation
The following example instances one CAPTURE object
CAPTURE cap1;

Initialization
To Instance pre-initialized objects
CAPTURE cap1 = CAPTURE_DEFAULTS;

Invoking the computation function
cap1.init(&cap1);
cap1.read(&cap1);

Example

The following pseudo code provides the information about the module usage.

```
main()
{

        cap1.init(&cap1);                   // Call init function for cap1

}


void interrupt periodic_interrupt_isr()
{
   Uint16 Status;
   Uint32 time_of_event;

   status = cap1.read(&cap1);                   // Call the capture read function

   // if status==1 then a time stamp was not read,
   //   if status==0 then a time stamp was read

   if(status==0)
   {
      time_of_event  = (Uint32)(cap1.TimeStamp);        // Read out new time stamp
   }

}
```