

Event Engine

What is the event engine?

- Software to run Enabler platforms independently without external controller
- Can be triggered by various input events
- Can execute various output events
- No development tools required
- Easy to program via AT command
- Up to 150 events programmable
- Event groups can be defined which run independently from other groups

Event Engine

Event Processing

Type of input events:

- **Power On** (8)
- GSM / GPRS Registration (9,10)
- GPIO state (0-7, 73-84)
- Receipt of Network IP (11)
- ADC Input (18)
- RTC alarm input (28)
- Counter (51)
- SMS Receipt (52)
- Timer (12-15, 66-69)

Type of output events:

- UDP message generation (40-42)
- Set GPO to low or high (8-23)
- Drive/flash output line (24-39)
- **Execute stored AT commands** (44)
- Reset or delete event timers (43)
- Set RTC ON/OFF times (53)
- Send SMS message (45)
- Event counter threshold (47)
- Change GPIO from output to input (1-7)
- Generate TCP/IP message (52)

Event Engine

Event Processing

Type of input events:

- Power On (8)
- GSM / GPRS Registration (9,10)
- GPIO state (0-7, 73-84)
- Receipt of Network IP (11)
- **ADC Input** (18)
- RTC alarm input (28)
- Counter (51)
- **SMS Receipt** (52)
- Timer (12-15, 66-69)

Type of output events:

- **UDP message generation** (40-42)
- Set GPO to low or high (8-23)
- Drive/flash output line (24-39)
- Execute stored AT commands (44)
- Reset or delete event timers (43)
- Set RTC ON/OFF times (53)
- **Send SMS message** (45)
- Event counter threshold (47)
- Change GPIO from output to input (1-7)
- Generate TCP/IP message (52)

Event Engine

Syntax

AT\$EVENT=<event group>,
 <event type>,
 <event category>,
 <parm1>,
 <parm2>

<event group> group of events containing one input and one output event at least
<event type> input (transition, occurrence) or output event
<event category> type of input/output event
<parm1> to be set in dependency of event category
<parm2> to be set in dependency of event category

Event Engine

<event type>

0 = Input event type “Transition”

1 = Input event type “Occurrence”

2 = logical AND

(e.g. “timer expiration” AND “valid IP address”)

3 = Output event

Input - Transition

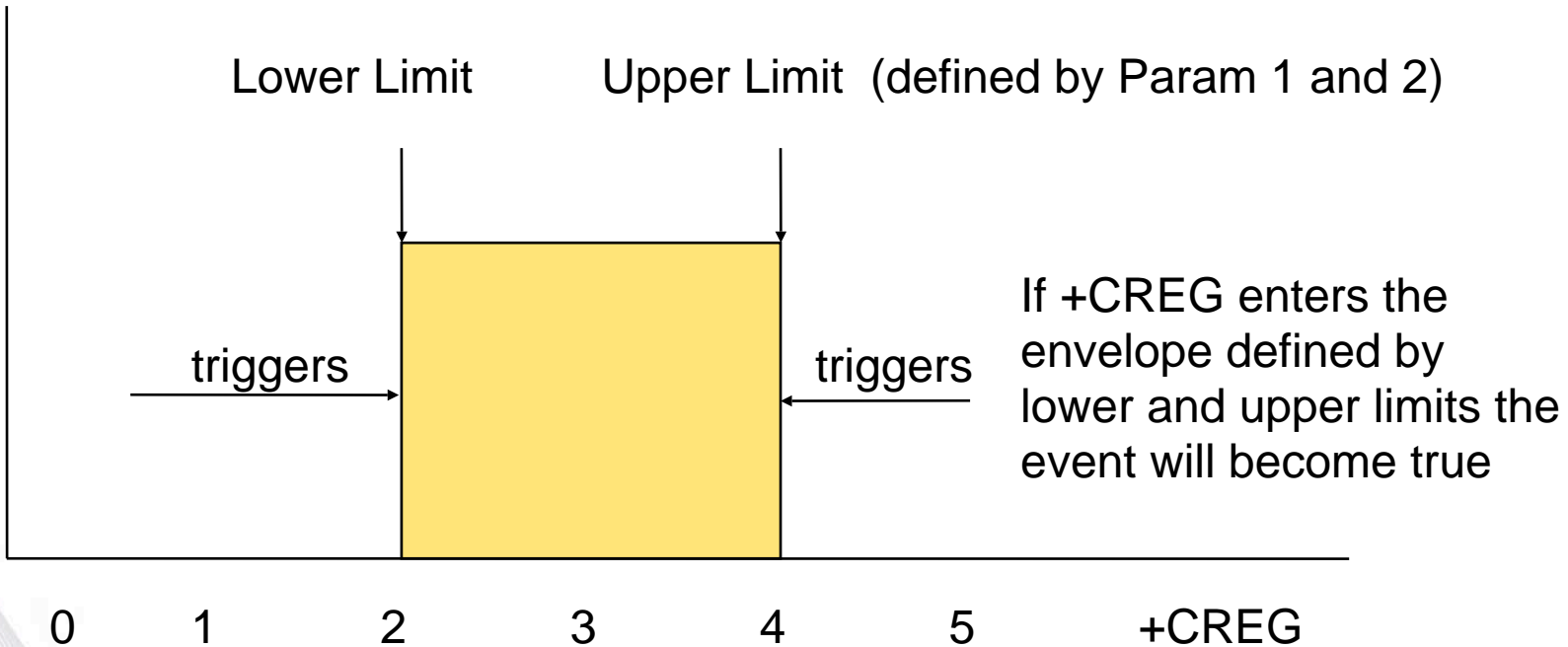
- Triggers only when the input becomes true.
- Will not re-trigger until the input has become false first before becoming true again
- Param 1 and 2 define the upper and lower limits of the input

Input – Transition contd.

Example:

- Input is GSM Registration status
- Possible values 0, 1, 2, 3, 4 and 5
- Shall trigger when input is 2,3 or 4 (not registered)
- Lower limit is 2 and upper limit is 4.
- Triggers event as soon as CREG is 2, 3 or 4
- Will not re-trigger if CREG changes between 2, 3 or 4
- Must first becomes 0,1 or 5 before triggers again

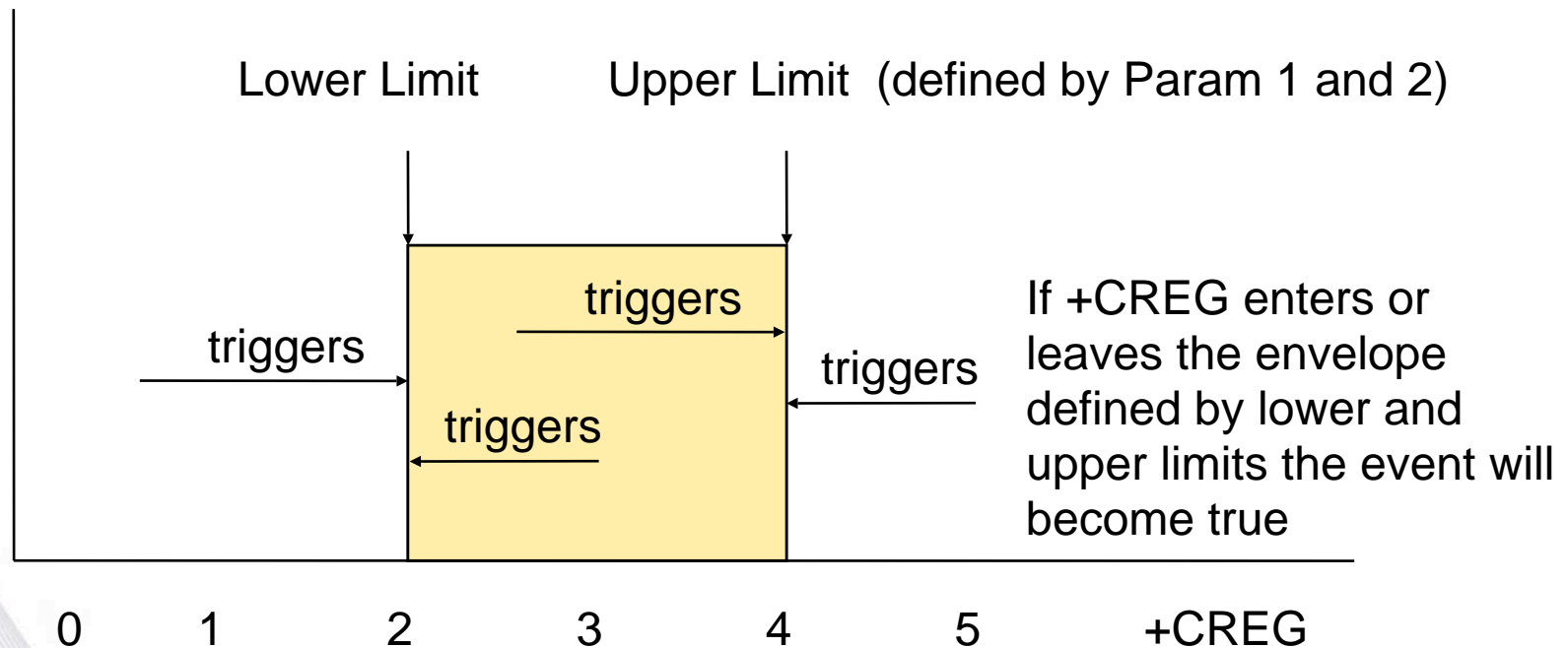
Example: Transition



Input - Occurrence

- Triggers and re-triggers always when the input becomes true
- Param 1 and 2 define the upper and lower limits of the input

Example: Occurrence



Input – Occurrence contd.

- Many Inputs are software updated (timer, GPS fix) so will re-trigger each time the software updates them if occurrence is selected and the value remains true.

Inputs – AND

- For an event to ‘fire’, it must have one of the 2 types of trigger already mentioned.
- Once this event has ‘triggered’, it can be ANDed with any other event – the output will only be set if the ANDed condition is true when the input is fired.
- If the input event triggers and the ANDed event is false, the input conditions will not be tested again until the input trigger re-occurs.

Event Engine

<event type> Example Transition Event

AT\$EVENT=23,0,12,1,1

/* If Timer 1 expires

AT\$EVENT=23,1,1,1,1

/* AND if GPIO2 **GOES** high

AT\$EVENT=23,3,40,333,4350

/* send UDP message

<event type> Example Occurrence Event

AT\$EVENT=24,0,12,1,1

/* If Timer 1 expires

AT\$EVENT=24,0,1,1,1

/* AND if GPIO2 **IS** high

AT\$EVENT=24,3,40,333,4350

/* send UDP message

Example 1. - Serial Port Event

Send message over the serial port when IP address is lost or has changed

```
AT+CGDCONT=1,"IP","MYAPN","",0,0
```

```
AT$AREG=1
```

```
AT$STOATEV=1,AT$MSGSEND=0,"IP Obtained"
```

```
AT$STOATEV=2,AT$MSGSEND=0,"IP Lost"
```

```
AT$EVENT=5,1,11,1,1
```

/* input event watch for new IP address during GPRS context activation

```
AT$EVENT=5,3,44,1,0
```

/* output event execute 1. stored AT cmd

```
AT$EVENT=6,1,11,0,0
```

/* input event watch for loss of an IP address

```
AT$EVENT=6,3,44,2,0
```

/* output event execute 2nd stored AT cmd

Example 2. - TCP connect on SMS

Facilitate a TCP PAD connection upon receipt of an SMS message

Process flow:

1. Modem receives SMS
2. IP address and port are predefined
3. Modem sends UDP messages with MDMID name to server for identification and processing
4. The TCP connection will “time out” if a connection is not made within 60 seconds
5. The TCP connection will disconnect after 30 seconds if no data being sent over the link
6. Modem will be configured once and settings are saved to non-volatile memory

Example 3. - TCP connect on SMS

Modem Configuration:

AT+CGDCONT=1,"IP","MYAPN","",0,0 /* Context activation

AT\$AREG=1 /* Auto registration for GPRS

AT\$ACTIVE=1 /* Defines Modem as TCP client

AT\$HOSTIF=2 /* Set host interface for TCP processing

AT\$STOATEV=1,ATD*99# /* AT command to establish a data connection

AT\$PADDST="62.126.72.126",1725 /* Server address modem connects to
when ATD*99# is dialed

AT\$UDPAPI=,1721 /* port used for UDPAPI

Example 3. - TCP connect on SMS

Modem Configuration:

AT\$CONNTO=60 /* no. of seconds the modem tries to
connect to server. If no success modem will go back to default
GSM
mode

AT\$IDLETO=30 /* seconds to wait before terminating a
TCP connection if no data has passed

AT\$FRIEND=1,1," 62.126.72.126" /* Friend server for UDP messages
Port from UDPAPI command is used

AT\$MDMID="TCPTTEST" /* Modem name

AT&W /* write settings to non volatile memory

Example 3. - TCP connect on SMS

Modem Event Configuration:

AT\$EVENT=5,1,52,1,1 /* Watch for inbound SMS to process it as an input event

AT\$EVENT=5,3,44,1,0 /* Execute the stored AT command ATD*99#
A TCP PAD connection to IP
62.126.72.126, port 1725 will be established

AT\$EVENT=5,3,42,0,4 /* Generate UDP messages to ALL IP
addresses in the AT\$FRIEND list (port from
UDPAPI is used)

Example 3. – Monitor GPRS State

Reset modem after 10 minutes if GPRS and GSM registration status are not valid

Bad condition

```
AT$EVENT=10,0,10,0,0          /* If no GPRS registration (%CGREG IS 0)
AT$EVENT=10,3,43,1,600       /* Set timer 1 to 10 minutes
AT$EVENT=20,0,9,2,4          /* or if CREG is between 2 and 5
AT$EVENT=20,3,43,1,600       /* Set timer 1 to 10 minutes
```

Good condition

```
AT$EVENT=30,0,10,1,9         /* If %CGREG is between 1 to 9           (everything is
good but 0)
AT$EVENT=30,2,9,1,1          /* AND +CREG=1 (roaming 5 is not considered
here)
AT$EVENT=30,3,43,1,0         /* stop timer 1
```

Action if timer expires

```
AT$EVENT=40,1,12,1,1        /* If timer1 expires
AT$EVENT=40,3,44,1,0        /* Execute STOATEV AT command in slot1
```

Argument for “Stored AT command event”

```
AT$STOATEV=1,AT$RESET       /* Stored AT command “Reset Modem”
```

Event Engine on GenTrack 11e

Type of input events:

- Button pressed (58)
- GPS Distance (16)
- Maximum Velocity (17)
- GPS Status (27)
- Motion status (62)
- Leaving, entering geo fence (21)
- Battery level (59)
- Power source (63)
- LTO download indicator (64)
-

Type of output events:

- Set Geo Fence (49)
- Send data via SMS (54 - 58)
- Turn off modem (59)
- Send GPS data according to settings via TCP and UDP (40-42, 52)
- etc.
- all output events from GSM0108

Reporting GPS position to a server

Report GPS position if > button is pressed

Basic Settings

```
AT+CGDCONT=1,"IP","internet"
```

```
AT%CGPCO=1,"xxx,xxx",1
```

```
AT$UPDAPI=,1721
```

```
AT$FRIEND=1,1,"apitest.enfora.com"
```

```
AT$MDMID=,,ENFORA-TEST "
```

```
AT$WAKEUP=1,1
```

```
AT$HOSTIF=1
```

```
AT$AREG=2
```

Reporting GPS position to a server

Event Processing:

AT\$EVENT=10,1,58,1,1 /* User defined button event

AT\$EVENT=10,3,40,333,4350 /* send NMEA message to Server

Result:

Everytime the user defined button (“<”) is pressed, a UDP message is sent to the first address listed in AT\$FRIEND using the port from UDPAPI.

<http://apitest.enfora.com>

Enter MDMID name (“ENFORA-TEST”)

```
$GPRMC,150010.00,A,5000.315882,N,02101.977782,E,  
0.0,0.0,010508,,A*51
```

Conversion of NMEA data

\$GPRMC,150010.00,A,5000.315882,N,02101.977782,E,0.0,0.0,010508,,,A*51

NMEA data format: XXYY.ZZZZ ,N,XXXYY.ZZZZ

Conversion into degree, minutes and seconds:

$$\text{XXYY.ZZZZ} = \text{XX} + (\text{YYZZZZ} / 600000)$$

Latitude 5000.3158 = 50 + (003158 / 600000)

+50.005263

Longitude 2101.9777 = 21 + (019777 / 600000)

+21.032967

W (west) and S (south) -> negativ prefix

Letting the GenTrack 11e hibernate

GenTrack 11e can be transferred into hibernation to save power during the time the device is not used

- When no motion or on demand
- Wakes up frequently or on motion
- During hibernation only the MSP430 waits in low power mode for any trigger to be woken up

Programming Example:

- GenTrack 11e goes to sleep when there is no motion for 5 minutes
- It will wake up on motion
- Will send it's position on wakeup
- If no valid position can be fixed it will send the last known one
- GenTrack 11e will start regarding "stop" after 1 minute

Programming Example:



Settings for hibernation

```
AT$WAKEENBL=4          /* Modem will wake up on motion
AT$MOTTRANS=60         /* stay in motion for 60 seconds
AT$WAKETIME=300        /* go to sleep after 5 minutes if stop is detected
```

Event machine settings

```
AT$EVENT=10,1,11,1,1   /* new IP received after GenTrack 11e woke up
AT$EVENT=10,3,40,999, 528638 /* send NMEA data via UDP
```