21 LESENSE - Low Energy Sensor Interface

**Quick Facts**

**What?**
LESENSE is a low energy sensor interface capable of autonomously collecting and processing data from multiple sensors even when in EM2. Flexible configuration makes LESENSE a versatile sensor interface compatible with a wide range of sensors and measurement schemes.

**Why?**
Capability to autonomously monitor sensors allows the EFM32TG to reside in a low energy mode for long periods of time while keeping track of sensor status and sensor events.

**How?**
LESENSE is highly configurable and is capable of collecting data from a wide range of sensor types. Once the data is collected, the programmable state machine, LESENSE decoder, is capable of processing sensor data without CPU intervention. A large result buffer allows the chip to remain in EM2 for long periods of time while autonomously collecting data.

### 21.1 Introduction

LESENSE is a low energy sensor interface which utilizes on-chip peripherals to perform measurement of a configurable set of sensors. The results from sensor measurements can be processed by the LESENSE decoder, which is a configurable state machine with up to 16 states. The results can also be stored in a result buffer to be collected by CPU or DMA for further processing.

LESENSE operates in EM2, in addition to EM1 and EM0, and can wake up the CPU on configurable events.

### 21.2 Features

- Up to 16 sensors
- Autonomous sensor monitoring in EM0, EM1, and EM2
- Highly configurable decoding of sensor results
- Interrupt on sensor events
- Configurable enable signals to external sensors
- Circular buffer for storage of up to 16 sensor results.
- Support for multiple sensor types
  - LC sensors
  - Capacitive sensing
  - General analog sensors
21.3 Functional description

LESENSE is a module capable of controlling on-chip peripherals in order to perform monitoring of different sensors with little or no CPU intervention. LESENSE uses the analog comparators, ACMP, for measurement of sensor signals. LESENSE can also control the DAC to generate accurate reference voltages. Figure 21.1 (p. 318) shows an overview of the LESENSE module. LESENSE consists of a sequencer, count and compare block, a decoder, and a RAM block used for configuration and result storage. The sequencer handles interaction with other peripherals as well as timing of sensor measurements. The count and compare block is used to count pulses from ACMP outputs before comparing with a configurable threshold. To autonomously analyze sensor results, the LESENSE decoder provides possibility to define a finite state machine with up to 16 states, and programmable actions upon state transitions. This allows the decoder to implement a wide range of decoding schemes, for instance quadrature decoding. A RAM block is used for storage of configuration and measurement results. This allows LESENSE to have a relatively large result buffer enabling the chip to remain in a low energy mode for long periods of time while collecting sensor data.

Figure 21.1. LESENSE block diagram

21.3.1 Channel configuration

LESENSE has 16 individually configurable channels, the first eight are mapped to the channels of ACMP0, while the last eight are mapped to the channels of ACMP1. Each LESENSE channel has its own set of configuration registers. Channel configuration is split into three registers; CHx_TIMING, CHx_INTERACT, and CHx_EVAL. Individual timing for each sensor is configured in CHx_TIMING, sensor interaction is configured in CHx_INTERACT, and configurations regarding evaluation of the measurements are done in CHx_EVAL. For improved readability, CHx_CONF will be used to address
the channel configuration registers, CHx_TIMING, CHx_INTERACT, and CHx_EVAL, throughout this chapter.

By default, the channel configuration registers are directly mapped to the channel number. Configuring SCANCONF in CTRL makes it possible to alter this mapping.

Configuring SCANCONF to INVMAP will make channels 0-7 use the channel configuration registers for channels 8-15, and vice versa. This feature allows an application to quickly and easily switch configuration set for the channels.

Setting SCANCONF to TOGGLE will make channel x alternate between using CHx_CONF and CHx+8_CONF. The configuration used is decided by the state of the corresponding bit in SCANRES. For instance, if channel 3 is performing a scan and bit 3 in SCANRES is set, CH11_CONF will be used. Channels 8 through 15 will toggle between CHx_CONF and CHx-8_CONF. This mode provides an easy way for implementation of hysteresis on channel events as threshold values can be changed depending on sensor status.

Setting SCANCONF to DECDEF will make the state of the decoder define which scan configuration to be used. If the decoder state is at index 8 or higher, channel x will use CHx+8_CONF, otherwise it will use CHx configuration. Similarly, channels 8 through 15 will use CHx configuration when decoder state index is less than 8 and CHx-8_CONF when decoder state index is higher than 7. Allowing the decoder state to define which configuration to use, enables easy implementation of for instance hysteresis, as different threshold values can be used for the same channel, depending on the state of the application. Table 21.1 (p. 319) summarizes how channel configuration is selected for different setting of SCANCONF.

### Table 21.1. LESENSE scan configuration selection

<table>
<thead>
<tr>
<th>LESENSE channel x</th>
<th>SCANCONF</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIRMAP</td>
<td>INVMAP</td>
</tr>
<tr>
<td>SCANCONF</td>
<td>TOGGLE</td>
</tr>
<tr>
<td>SCANRES[n] = 0</td>
<td>SCANRES[n] = 1</td>
</tr>
<tr>
<td>0 &lt;= x &lt; 8</td>
<td>CHx_CONF</td>
</tr>
<tr>
<td>8 &lt;= x &lt; 16</td>
<td>CHx_CONF</td>
</tr>
</tbody>
</table>

Channels are enabled in the CHEN register, where bit x enables channel x. During a scan, all enabled channels are measured, starting with the lowest indexed channel. Figure 21.2 (p. 320) illustrates a scan sequence with channels 3, 5, and 9 enabled.

#### 21.3.2 Scan sequence

LESENSE runs on LFACLK\_LESENSE, which is a prescaled version of LFACLK. The prescaling factor for LFACLK\_LESENSE is selected in the CMU, available prescaling factors are:

- DIV1: LFACLK\_LESENSE = LFACLK/1
- DIV2: LFACLK\_LESENSE = LFACLK/2
- DIV4: LFACLK\_LESENSE = LFACLK/4
- DIV8: LFACLK\_LESENSE = LFACLK/8

**Note**

LFACLK\_LESENSE should not exceed 50kHz.

All enabled channels are scanned each scan period. How a new scan is started is configured in the SCANMODE bit field in CTRL. If set to PERIODIC, the scan frequency is generated using a counter which
is clocked by LFACLK_LESENSE. This counter has its own prescaler. This prescaling factor is configured in PCPRESC in TIMCTRL. A new scan sequence is started each time the counter reaches the top value, PCTOP. The scan frequency is calculated using Equation 21.1 (p. 320). If SCANMODE is set to ONESHOT, a single scan will be made when START in CMD is set. To start a new scan on a PRS event, set SCANMODE to PRS and configure PRS channel in PRSSEL. The PRS start signal needs to be active for at least one LFACLK_LESENSE cycle to make sure LESENSE is able to register it.

Scan frequency

\[ F_{\text{scan}} = \frac{\text{LFACLK}_{\text{LESENSE}}}{(1 + \text{PCTOP}) \cdot 2^{\text{PCPRESC}}} \]  

(21.1)

It is possible to interleave additional sensor measurements in between the periodic scans. Issuing a start command when LESENSE is idle will immediately start a new scan, without disrupting the frequency of the periodic scans. If the period counter overflows during the interleaved scan, the periodically scheduled scan will start immediately after the interleaved scan completes.

**Figure 21.2. Scan sequence**

**21.3.3 Sensor timing**

For each channel in the scan sequence, the LESENSE interface goes through three phases: Idle phase, excite phase, and measure phase. The durations of the excite and measure phases are configured in the CHx_TIMING registers. Timing of the excite phase can be either a number of AUXHFRCO cycles or a number of LFACLK_LESENSE cycles, depending on which one is selected in EXCLK. LESENSE includes two timers: A low frequency timer, running on LFACLK_LESENSE, and a high frequency timer, running on AUXHFRCO. The low frequency timer can be prescaled by configuring LFPRESC in TIMCTRL, and the high frequency timer prescaling factor is configured in AUXPRES in the same register. The duration of the measure phase is programmed via MEASUREDLY and SAMPLEDLY. The output of the ACMP will be inactive for MEASUREDLY EXCLK cycles after start of the sensor measurement. Sampling of the sensor will happen after SAMPLEDLY LFACLK_LESENSE, or AUXHFRCO cycles, depending on the configuration of SAMPLECLK. Figure 21.3 (p. 320) depicts a sensor sequence where excitation and measure delay is timed using AUXHFRCO and the sample delay is timed using LFACLK_LESENSE. The configurable measure- and sample delays enables LESENSE to easily define exact time windows for sensor measurements. A start delay can be inserted before sensor measurement begin by configuring STARTDLY in TIMCTRL. This delay can be used to ensure that the DAC is done and voltages have stabilized before sensor measurement begins.

**Figure 21.3. Timing diagram, short excitation**
21.3.4 Sensor interaction

Many sensor types require some type of excitation in order to work. LESENSE can generate a variety of sensor stimuli, both on the same pin as the measurement is to be made on, and on alternative pins.

By default, excitation is performed on the pin associated with the channel, i.e. excitation and sensor measurement is performed on the same pin. The mode of the pin during the excitation phase is configured in EXMODE in CHx_INTERACT. The available modes during the excite phase are:

- **DISABLED**: The pin is disabled.
- **HIGH**: The pin is driven high.
- **LOW**: The pin is driven low.
- **DACOUT**: The pin is connected to the output of a DAC channel.

**Note**

Excitation with DAC output is only available on channels 0, 1, 2, and 3 (DAC0_CH0) and channels 12, 13, 14, and 15 (DAC0_CH1).

If the DAC is in opamp-mode, setting EXMODE to DACOUT will result in excitation with output from the opamp.

LESENSE is able to perform sensor excitation on another pin than the one to be measured. When ALTEX in CHx_INTERACT is set, the excitation will occur on the alternative excite pin associated with the given channel. All LESENSE channels mapped to ACMP0 have their alternative channel mapped to the corresponding channel on ACMP1, and vice versa. Alternatively, the alternative excite pins can be routed to the LES_ALTEX pins. Mapping of the alternative excite pins is configured in ALTEXMAP in CTRL. Table 21.2 (p. 321) summarizes the mapping of excitation pins for different configurations.

**Table 21.2. LESENSE excitation pin mapping**

<table>
<thead>
<tr>
<th>LESENSE channel</th>
<th>ALTEX = 0</th>
<th>ALTEX = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ALTEXMAP = ACMP</td>
<td>ALTEXMAP = ALTEX</td>
</tr>
<tr>
<td>0</td>
<td>ACMP0_CH0</td>
<td>ACMP1_CH0</td>
</tr>
<tr>
<td>1</td>
<td>ACMP0_CH1</td>
<td>ACMP1_CH1</td>
</tr>
<tr>
<td>2</td>
<td>ACMP0_CH2</td>
<td>ACMP1_CH2</td>
</tr>
<tr>
<td>3</td>
<td>ACMP0_CH3</td>
<td>ACMP1_CH3</td>
</tr>
<tr>
<td>4</td>
<td>ACMP0_CH4</td>
<td>ACMP1_CH4</td>
</tr>
<tr>
<td>5</td>
<td>ACMP0_CH5</td>
<td>ACMP1_CH5</td>
</tr>
<tr>
<td>6</td>
<td>ACMP0_CH6</td>
<td>ACMP1_CH6</td>
</tr>
<tr>
<td>7</td>
<td>ACMP0_CH7</td>
<td>ACMP1_CH7</td>
</tr>
<tr>
<td>8</td>
<td>ACMP1_CH0</td>
<td>ACMP0_CH0</td>
</tr>
<tr>
<td>9</td>
<td>ACMP1_CH1</td>
<td>ACMP0_CH1</td>
</tr>
<tr>
<td>10</td>
<td>ACMP1_CH2</td>
<td>ACMP0_CH2</td>
</tr>
<tr>
<td>11</td>
<td>ACMP1_CH3</td>
<td>ACMP0_CH3</td>
</tr>
<tr>
<td>12</td>
<td>ACMP1_CH4</td>
<td>ACMP0_CH4</td>
</tr>
<tr>
<td>13</td>
<td>ACMP1_CH5</td>
<td>ACMP0_CH5</td>
</tr>
<tr>
<td>14</td>
<td>ACMP1_CH6</td>
<td>ACMP0_CH6</td>
</tr>
<tr>
<td>15</td>
<td>ACMP1_CH7</td>
<td>ACMP0_CH7</td>
</tr>
</tbody>
</table>
Figure 21.4 (p. 322) illustrates the sequencing of the pin associated with the active channel and its alternative excite pin.

**Figure 21.4. Pin sequencing**

The alternative excite pins, LES_ALTEXn, have the possibility to excite regardless of what channel is active. Setting AEXn in ALTEXCONF will make LES_ALTEXn excite for all channels using alternative excitation, i.e. ALTEX in CHx_INTERACT is set.

**Note**

When exciting on the pin associated with the active channel, the pin will go through a tristated phase before returning to the idle configuration. This will not happen on pins used as alternative excitation pins.

The pin configuration for the idle phase can be configured individually for each LESENSE channel and alternative excite pin in the IDLECONF and ALTEXCONF registers. The modes available are the same as the modes available in the excitation phase. In the measure phase, the pin mode on the active channel is always disabled (analog input).

To enable LESENSE to control GPIO, the pin has to be enabled in the ROUTE register. In addition, the given pin must be configured as push-pull. IDLECONF configuration should not be altered when pin enable for the given pin is set in ROUTE.

**21.3.5 Sensor evaluation**

Sensor evaluation can be based on either analog comparator outputs, or the counter output. This is configured in the SAMPLE bitfield in CHx_INTERACT. The LESENSE counter is used to count pulses on the ACMP output in the measurement phase. When a measurement phase is completed, the counter value is compared to the value configured in COMPTHRES in CHx_EVAL. By configuring COMP, it is possible to choose comparison mode: Less than, or greater than or equal. If a comparison for a channel triggers, the corresponding bit in the result register, SCANRES, is set. To set an interrupt flag on a sensor event, configure SETIF in CHx_INTERACT. Figure 21.5 (p. 323) illustrates how the counter value or ACMP sample is used for evaluation and interrupt generation.
LESENSE includes the possibility to sample both analog comparators simultaneously, effectively cutting the time spent on sensor interaction in some applications in half. Setting DUALSAMPLE in CTRL enables this mode. In dual sample mode, the channels of ACMP0 are paired together with the corresponding channel on ACMP1, i.e. channel x on ACMP0 and channel x on ACMP1 are sampled simultaneously. The results from sensor measurements can be fed into the decoder register and/or stored in the result buffer. In this mode, the samples from the AMCPs are placed in the two LSBs of the result stored in the result buffer. Results from both ACMPs will be evaluated for interrupt generation.

### 21.3.6 Decoder

Many applications require some sort of processing of the sensor readings, for instance in the case of quadrature decoding. In quadrature decoding, the sensors repeatedly pass through a set of states which corresponds to the position of the sensors. This sequence, and many other decoding schemes, can be described as a finite state machine. To support this type of decoding without CPU intervention, LESENSE includes a highly configurable decoder, capable of decoding input from up to four sensors. The decoder is implemented as a programmable state machine with up to 16 states. When doing a sensor scan, the results from the sensors are placed in the decoder input register, SENSORSTATE, if DECODE in CHx_INTERACT is set. The resulting position after a scan is illustrated in Figure 21.6 (p. 323), where the bottom blocks show how the SENSORSTATE register is filled. When the scan sequence is complete, the decoder evaluates the state of the sensors chosen for decoding, as depicted in Figure 21.6 (p. 323).

**Figure 21.6. Sensor scan and decode sequence**

The decoder is a programmable state machine with support for up to 16 states. The behavior of each state is individually configured in the STx_TCONFA and STx_TCONFb registers. The registers define possible transitions from the present state. If the sensor state matches COMP in either STx_TCONFA or STx_TCONFb, a transition to the state defined in NEXTSTATE will be made. It is also possible to mask out one or more sensors using the MASK bit field. The state of a masked sensor is interpreted as don’t care.
Upon a state transition, LESENSE can generate a pulse on one or more of the decoder PRS channels. Which channel to generate a pulse on is configured in the PRSACT bit field. If PRSCNT in DECCTRL is set, count signals will be generated on decoder PRS channels 0 and 1 according to the PRSACT configuration. In this mode, channel 0 will pulse each time a count event occurs while channel 1 indicates the count direction, 1 being up and 0 being down. The count direction will be kept at its previous state in between count events. The EFM32TG pulse counter may be used to keep track of events based on these PRS outputs.

If SETIF is set, the DECODER interrupt flag will be set when the transition occurs. If INTMAP in DECCTRL and SETIF is set, a transition from state x will set the CHx interrupt flag in addition to the DECODER flag.

Setting CHAIN in STx_TCONF enables the decoder to evaluate more than two possible transitions for each state. If none of the transitions defined in STx_TCONF or STx_TCONF matches, the decoder will jump to the next descriptor pair and evaluate the transitions defined there. The decoder uses two LFACLK cycles for each descriptor pair to be evaluated. If ERRCHK in CTRL is set, the decoder will check that the sensor state has not changed if none of the defined transitions match. The DECERR interrupt flag will be set if none of the transitions match and the sensor state has changed. Figure 21.7 (p. 324) illustrates state transitions. The "Generate PRS signals and set interrupt flag" blocks will perform actions according to the configuration in STx_TCONF and STx_TCONF.

**Figure 21.7. Decoder state transition evaluation**
Note
If only one transition from a state is used, STx_TCONFA and STx_TCONFB should be configured equally.

To prevent unnecessary interrupt requests or PRS outputs when the decoder toggles back and forth between two states, a hysteresis option is available. The hysteresis function is triggered if a type A transition is preceded by a type B transition, and vice versa. A type A transition is a transition defined in STx_TCONFA, and a type B transition is a transition defined in STx_TCONFB. When descriptor chaining is used, a jump to another descriptor will cancel out the hysteresis effect. Figure 21.8 (p. 325) illustrates how the hysteresis triggers upon state transitions.

Figure 21.8. Decoder hysteresis

The events suppressed by the hysteresis are configured in bit fields HYSTPRS0-2 and HYSTIRQ in DECCCTRL.

• When HYSTPRSx is set, PRS signal x is suppressed when the hysteresis triggers.
• When HYSTIRQ is set, interrupt requests are suppressed when the hysteresis triggers.

Note
The decoder error interrupt flag, DECERR, is not affected by the hysteresis.

21.3.7 Measurement results

Part of the LESENSE RAM is treated as a circular buffer for storage of up to 16 results from sensor measurements. Each time LESENSE writes data to the result buffer, the result write pointer, PTR_WR, is incremented. Each time a new result is read through the BUFDATA register, the result read pointer, PTR_RD, is incremented. The read pointer will not be incremented if there is no valid, unread data in the result buffer. By default LESENSE will not write additional data to a full result buffer until the data is read by software or DMA. Setting BUFOW in CTRL enables LESENSE to write to the result buffer, even if it is full. In this mode, the result read pointer will follow the write pointer if the buffer is full. The result of this is that data read from the result read register, BUFDATA, is the oldest unread result. The location pointers are available in PTR. The result buffer has three status flags; BUFVALID, BUFHALF, and BUFFULL. The flags indicate when new data is available, when the buffer is half full, and when it is full, respectively. The interrupt flag BUFVALID is set when data is available in the buffer. BUFLEVEL is set when the buffer is either full or half full, depending on the configuration of BUFIDL in CTRL. If the result buffer overflows, the BUFOF interrupt flag will be set.

During a scan, the state of each sensor is stored in SCANRES. If a sensor triggers, a 1 is stored in SCANRES, else a 0 is stored in SCANRES. Whether or not a sensor is said to be triggered depends of the configuration for the given channel. If SAMPLE is set to ACMP, the sensor is said to be triggered if the output from the analog comparator is 1 when sensor sampling is performed. If SAMPLE is set to COUNTER, a sensor is said to be triggered if the LESENSE counter value is greater than or equal, or
less than COMPTHRES, depending on the configuration of COMP. If STRSAMPLE in CHx_EVAL is set, the counter value or ACM sample for each channel will be stored in the LESENSE result buffer. If STRSCANRES in CTRL is set, the result vector, SCANRES, will also be stored in the result buffer. This will be stored after each scan and will be interleaved with the counter values. The contents of the result buffer can be read from BUFDATA or from BUF[x]_DATA. When reading from BUF[x]_DATA, neither the result read pointer or the status flags BUFDATAV, BUFHALFFULL, or BUFFULL will be updated. When reading through the BUFDATA register, the oldest unread result will be read.

**Figure 21.9. Circular result buffer**

The right hand side of Figure 21.9 (p. 326) illustrates how the result buffer would be filled when channels 3, 5, and 9 are enabled and have STRSAMPLE in CHx_EVAL set, in addition to STRSCANRES in CTRL. The measurement result from the three channels will be sequentially written during the scan, while SCANRES is written to the result buffer upon scan completion.

### 21.3.8 DAC interface

LESENSE is able to drive the DAC for generation of accurate reference voltages. DAC channels 0 and 1 are individually configured in the PERCTRL register. The conversion mode can be set to either continuous, sample/hold or sample/off. For further details about these modes, refer to Section 25.3.1 (p. 406). Both DAC channels are refreshed prior to each sensor measurement, as depicted in Figure 21.3 (p. 320). The conversion data is either taken from the data registers in the EFM32TG DAC interface (DAC0_CH0DATA and DAC0_CH1DATA) or from the ACMPTHRES bitfield in the CHx_INTERACT register for the active LESENSE channel. DAC data used is configured in DACCHxDATA in PERCTRL.

The DAC interface runs on AUXHFRCO and will enable this when it is needed. The DACPRESC bitfield in PERCTRL is used to prescale the AUXHFRCO to achieve wanted clock frequency for the LESENSE DAC interface. The frequency should not exceed 1MHz. The prescaler may also be used to tune how long the DAC should drive its outputs in sample/off mode.

Bias configuration, calibration and reference selection is done in the EFM32TG DAC module and LESENSE will not override these configurations. If a bandgap reference is selected for the DAC, the DACREF bit in PERCTRL should be set to BANDGAP.

LESENSE has the possibility to control switches that connect the DAC outputs to the pins associated with ACMP0_CH0-3 and ACMP1_CH12-15. This makes LESENSE able to excite sensors with output from the DAC channels.

The DAC may be chosen as reference to the analog comparators for accurate reference generation. If the DAC is configured in continuous or sample/hold mode this does not require any external components. If sample/off mode is used, an external capacitor is needed to keep the voltage in between samples. To connect the input from the DAC to the ACMP to this external capacitor, connect the capacitor to the DAC pin for the given channel and set OPAxSHORT in DAC_OPACTRL.

**Note**

The DAC mode should not be altered while DACACTIVE in STATUS is set.
21.3.9 ACMP interface

The ACMPs are used to measure the sensors, and have to be configured according to the application in order for LESENSE to work properly. Depending on the configuration in the ACMP0MODE and ACMP1MODE bitfields in PERCTRL, LESENSE will take control of the positive input mux and the Vdd scaling factor (VDDLEVEL) for ACMP0 and ACMP1. The remaining configuration of the analog comparators are done in the ACMP register interface. It is recommended to set the MUXEN bit in ACMPn_CTRL for the ACMPs used by LESENSE. Each channel has the possibility to control the value of the Vdd scaling factor on the negative input of the ACMP, VDDLEVEL in ACMP_INPUTSEL. This is done in the 6 LSBs of ACMPTHRES in CHx_INTERACT. LESENSE automatically controls the ACMPmux to connect the correct channel.

21.3.10 ACMP and DAC duty cycling

By default, the analog comparators and DAC are shut down in between LESENSE scans to save energy. If this is not wanted, WARMUPMODE in PERCTRL can be configured to prevent them from being shut down.

Both the DAC and analog comparators rely on a bias module for correct operation. This bias module has a low power mode which consumes less energy at the cost of reduced accuracy. BIASMODE in BIASCTRL configures how the bias module is controlled by LESENSE. When set to DUTYCYCLE, LESENSE will set the bias module in high accuracy mode whenever LESENSE is active, and keep it in the low power mode otherwise. When BIASMODE is set to HIGHACC, the high accuracy mode is always selected. When set to DONTTOUCH, LESENSE will not control the bias module.

21.3.11 DMA requests

LESENSE issues a DMA request when the result buffer is either full or half full, depending on the configuration of BUFIDL in CTRL. The request is cleared when the buffer level drops below the threshold defined in BUFIDL. A single DMA request is also set whenever there is unread data in the buffer. DMAWU in CTRL configures at which buffer level LESENSE should wake up the DMA when in EM2.

Note

The DMA controller should always fetch data from the BUFDATA register.

21.3.12 PRS output

LESENSE is an asynchronous PRS producer and has nineteen PRS outputs. The decoder has three outputs and in addition, all bits in the SCANRES register are available as PRS outputs. For further information on the decoder PRS output, refer to Section 21.3.6 (p. 323).

21.3.13 RAM

LESENSE includes a RAM block used for storage of configuration and results. If LESENSE is not used, this RAM block can be powered down eliminating its current consumption due to leakage. The RAM is powered down by setting the RAM bit in the POWERDOWN register. Once the RAM has been shut down it cannot be turned back on without a reset of the chip. Registers mapped to the RAM include: STx_TCONF, STx_TCONFb, BUFx_DATA, BUFDATA, CHx_TIMING, CHx_INTERACT, and CHx_EVAL. These registers have unknown value out of reset and have to be initialized before use.

Note

Read-modify-write operations on uninitialized RAM register produces undefined values.

21.3.14 Application examples

21.3.14.1 Capacitive sense

Figure 21.10 (p. 328) illustrates how the EFM32TG can be configured to monitor four capacitive buttons.
The following steps show how to configure LESENSE to scan through the four buttons 100 times per second, issuing an interrupt if one of them is pressed.

1. Assuming LFACLK\textsubscript{LESENSE} is 32kHz, set PCPRESC to 3 and PCTOP to 39 in CTRL. This will make the LESENSE scan frequency 100Hz.
2. Enable channels 0 through 3 in CHEN and set IDLECONF for these channels to DISABLED. In capacitive sense mode, the GPIO should always be disabled (analog input).
3. Configure the ACMP to operate in CAPSENSE mode, refer to Section 22.3.5 (p. 366) for details.
4. Configure the following bit fields in CHx\_CONF, for channels 0 through 3:
   a. Set EXTIME to 0. No excitation is needed in this mode.
   b. Set SAMPLE to COUNTER and COMP to LESS. This makes LESENSE interpret a sensor as active if the frequency on a channel drops below the threshold, i.e. the button is pressed.
   c. Set SAMPLEDLY to an appropriate value, each sensor will be measured for SAMPLEDLY/ LFACLK\textsubscript{LESENSE} seconds. MEASUREDLY should be set to 0
5. Set CTRTHRESHOLD to an appropriate value. An interrupt will be issued if the counter value for a sensor is below this threshold after the measurement phase.
6. Enable interrupts on channels 0 through 3.
7. Start scan sequence by writing a 1 to START in CMD.

In a capacitive sense application, it might be required to calibrate the threshold values on a periodic basis, this is done in order to compensate for humidity and other physical variations. LESENSE is able to store up to 16 counter values from a configurable number of channels, making it possible to collect sample data while in EM2. When calibration is to be performed, the CPU only has to be woken up for a short period of time as the data to be processed already lies in the result registers. To enable storing of the count value for a channel, set STRSAMPLE in the CHx\_INTERACT register.

### 21.3.14.2 LC sensor

Figure 21.11 (p. 328) below illustrates how the EFM32TG can be set up to monitor four LC sensors.

![LC sensor setup](image)

LESENSE can be used to excite and measure the damping factor in LC sensor oscillations. To measure the damping factor, the ACMP can be used to generate a high output each time the sensor voltage exceeds a certain level. These pulses are counted using an asynchronous counter and compared with...
the threshold in COMPTHRES in the CHx_EVAL register. If the number of pulses exceeds the threshold level, the sensor is said to be active, otherwise it is inactive. Figure 21.12 (p. 329) illustrates how the output pulses from the ACMP correspond to damping of the oscillations. The results from sensor evaluation can automatically be fed into the decoder in order to keep track of rotations.

**Figure 21.12. LC sensor oscillations**

![LC sensor oscillations](image)

The following steps show how to configure LESENSE to scan through the four LC sensors 100 times per second.

1. Assuming LFACLK_LESENSE is 32kHz, set PCPRESC to 3 and PCTOP to 39 in CTRL. This will make the LESENSE scan frequency 100Hz.
2. Enable the DAC and configure it to produce a voltage of Vdd/2.
3. Enable channels 0 through 3 in CHEN. Set IDLECONF for the active channels to DACOUT. The channel pins should be set to Vdd/2 in the idle phase to damp the oscillations.
4. Configure the ACMP to use scaled Vdd as negative input, refer to ACMP chapter for details.
5. Enable and configure PCNT and asynchronous PRS.
6. Configure the GPIOs used as PUSHPULL.
7. Configure the following bit fields in CHx_CONF, for channels 0 through 3:
   a. Set EXCLK to AUXHFRCO. AUXHFRCO is needed to achieve short excitation time.
   b. Set EXTIME to an appropriate value. Excitation will last for EXTIME/AUXHFRCO seconds.
   c. Set EXMODE to LOW. The LC sensors are excited by pulling the excitation pin low.
   d. Set SAMPLE to COUNTER and COMP to LESS. Status of each sensor is evaluated based on the number of pulses generated by the ACMP. If they are less than the threshold value, the sensor is said to be active.
   e. Set SAMPLEDLY to an appropriate value, each sensor will be measured for SAMPLEDLY/LFACLK_LESENSE seconds.
8. Set CTRTHRESHOLD to an appropriate value. If the sensor is active, the counter value after the measurement phase should be less than the threshold. If it inactive, the counter value should be greater than the threshold.
9. Start scan sequence by writing a 1 to START in CMD.

### 21.3.14.3 LESENSE decoder 1

The example below illustrates how the LESENSE module can be used for decoding using three sensors

**Figure 21.13. FSM example 1**

![FSM example 1](image)

To set up the decoder to decode rotation using the encoding scheme seen in Figure 21.13 (p. 329), configure the following LESENSE registers:
1. Configure the channels to be used, be sure to set DECODE in CHx_EVAL.
2. Set PRSCNT to enable generation of count waveforms on PRS. Also configure a PCNT to listen to the PRS channels and count accordingly.
3. Configure the following in STx_TCONFA and STx_TCONFB:
   a. Set MASK = 0b1000 in STx_TCONFA and STx_TCONFB for all used states. This enables three sensors to be evaluated by the decoder.
   b. Configure the remaining bit fields in STx_TCONFA and STx_TCONFB as described in Table 21.3 (p. 330).

**Table 21.3. LESENSE decoder configuration**

<table>
<thead>
<tr>
<th>Register</th>
<th>TCONFA_NEXT</th>
<th>TCONFA_COMP</th>
<th>TCONFA_PRSACT</th>
<th>TCONFB_NEXT</th>
<th>TCONFB_COMP</th>
<th>TCONFB_PRSACT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST0</td>
<td>1</td>
<td>0b001</td>
<td>UP</td>
<td>7</td>
<td>0b100</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST1</td>
<td>2</td>
<td>0b011</td>
<td>UP</td>
<td>0</td>
<td>0b000</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST2</td>
<td>3</td>
<td>0b010</td>
<td>UP</td>
<td>1</td>
<td>0b001</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST3</td>
<td>4</td>
<td>0b110</td>
<td>UP</td>
<td>2</td>
<td>0b011</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST4</td>
<td>5</td>
<td>0b111</td>
<td>UP</td>
<td>3</td>
<td>0b010</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST5</td>
<td>6</td>
<td>0b101</td>
<td>UP</td>
<td>4</td>
<td>0b110</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST6</td>
<td>7</td>
<td>0b100</td>
<td>UP</td>
<td>5</td>
<td>0b111</td>
<td>DOWN</td>
</tr>
<tr>
<td>ST7</td>
<td>0</td>
<td>0b000</td>
<td>UP</td>
<td>6</td>
<td>0b101</td>
<td>DOWN</td>
</tr>
</tbody>
</table>

4. To initialize the decoder, run one scan, and read the present sensor status from SENSORSTATE. Then write the index of this state to DECSTATE.
5. Write to START in CMD to start scanning of sensors and decoding.

### 21.3.14.4 LESENSE decoder 2

The example below illustrates how the LESENSE decoder can be used to implement the state machine seen in Figure 21.14 (p. 330).

**Figure 21.14. FSM example 2**

1. Configure STx_TCONFA and STx_TCONFB as described in Table 21.4 (p. 331).
### Table 21.4. LESENSE decoder configuration

<table>
<thead>
<tr>
<th>Register</th>
<th>NEXTSTATE</th>
<th>COMP</th>
<th>MASK</th>
<th>CHAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST0_TCONFA</td>
<td>8</td>
<td>0b1000</td>
<td>0b0111</td>
<td>1</td>
</tr>
<tr>
<td>ST0_TCONFB</td>
<td>2</td>
<td>0b0001</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST1_TCONFA</td>
<td>6</td>
<td>0b0010</td>
<td>0b1000</td>
<td>0</td>
</tr>
<tr>
<td>ST1_TCONFB</td>
<td>6</td>
<td>0b0010</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST2_TCONFA</td>
<td>8</td>
<td>0b1000</td>
<td>0b0111</td>
<td>1</td>
</tr>
<tr>
<td>ST2_TCONFB</td>
<td>4</td>
<td>0b0011</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST3_TCONFA</td>
<td>0</td>
<td>0b0000</td>
<td>0b1000</td>
<td>0</td>
</tr>
<tr>
<td>ST3_TCONFB</td>
<td>0</td>
<td>0b0000</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST4_TCONFA</td>
<td>8</td>
<td>0b1000</td>
<td>0b0111</td>
<td>1</td>
</tr>
<tr>
<td>ST4_TCONFB</td>
<td>6</td>
<td>0b0010</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST5_TCONFA</td>
<td>2</td>
<td>0b0001</td>
<td>0b1000</td>
<td>0</td>
</tr>
<tr>
<td>ST5_TCONFB</td>
<td>2</td>
<td>0b0001</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST6_TCONFA</td>
<td>8</td>
<td>0b1000</td>
<td>0b0111</td>
<td>1</td>
</tr>
<tr>
<td>ST6_TCONFB</td>
<td>0</td>
<td>0b0000</td>
<td>0b1000</td>
<td>-</td>
</tr>
<tr>
<td>ST7_TCONFA</td>
<td>4</td>
<td>0b0011</td>
<td>0b1000</td>
<td>0</td>
</tr>
<tr>
<td>ST7_TCONFB</td>
<td>4</td>
<td>0b0011</td>
<td>0b1000</td>
<td>-</td>
</tr>
</tbody>
</table>

2. To initialize the decoder, run one scan, and read the present sensor status from SENSORSTATE. Then write the index of this state to DECSSATE.

3. Write to START in CMD to start scanning of sensors and decoding.
## 21.4 Register Map

The offset register address is relative to the registers base address.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x000</td>
<td>LESENSE_CTRL</td>
<td>RW</td>
<td>Control Register</td>
</tr>
<tr>
<td>0x004</td>
<td>LESENSE_TIMCTRL</td>
<td>RW</td>
<td>Timing Control Register</td>
</tr>
<tr>
<td>0x008</td>
<td>LESENSE_PERCTRL</td>
<td>RW</td>
<td>Peripheral Control Register</td>
</tr>
<tr>
<td>0x00C</td>
<td>LESENSE_DECCTRL</td>
<td>RW</td>
<td>Decoder control Register</td>
</tr>
<tr>
<td>0x010</td>
<td>LESENSE_BIASCTRL</td>
<td>RW</td>
<td>Bias Control Register</td>
</tr>
<tr>
<td>0x014</td>
<td>LESENSE_CMD</td>
<td>W1</td>
<td>Command Register</td>
</tr>
<tr>
<td>0x018</td>
<td>LESENSE_CHEN</td>
<td>RW</td>
<td>Channel enable Register</td>
</tr>
<tr>
<td>0x01C</td>
<td>LESENSE_SCANRES</td>
<td>R</td>
<td>Scan result register</td>
</tr>
<tr>
<td>0x020</td>
<td>LESENSE_STATUS</td>
<td>R</td>
<td>Status Register</td>
</tr>
<tr>
<td>0x024</td>
<td>LESENSE_PTR</td>
<td>R</td>
<td>Result buffer pointers</td>
</tr>
<tr>
<td>0x028</td>
<td>LESENSE_BUFDATA</td>
<td>R</td>
<td>Result buffer data register</td>
</tr>
<tr>
<td>0x02C</td>
<td>LESENSE_CURCH</td>
<td>R</td>
<td>Current channel index</td>
</tr>
<tr>
<td>0x030</td>
<td>LESENSE_DECSTATE</td>
<td>RWH</td>
<td>Current decoder state</td>
</tr>
<tr>
<td>0x034</td>
<td>LESENSE_SENSORSTATE</td>
<td>RWH</td>
<td>Decoder input register</td>
</tr>
<tr>
<td>0x038</td>
<td>LESENSE_IDLECONF</td>
<td>RW</td>
<td>GPIO idlephase configuration</td>
</tr>
<tr>
<td>0x03C</td>
<td>LESENSE_ALTEXCONF</td>
<td>RW</td>
<td>Alternative excite pin configuration</td>
</tr>
<tr>
<td>0x040</td>
<td>LESENSE_IF</td>
<td>R</td>
<td>Interrupt Flag Register</td>
</tr>
<tr>
<td>0x044</td>
<td>LESENSE_IFC</td>
<td>W1</td>
<td>Interrupt Flag Clear Register</td>
</tr>
<tr>
<td>0x048</td>
<td>LESENSE_IFS</td>
<td>W1</td>
<td>Interrupt Flag Set Register</td>
</tr>
<tr>
<td>0x04C</td>
<td>LESENSE_IEN</td>
<td>RW</td>
<td>Interrupt Enable Register</td>
</tr>
<tr>
<td>0x050</td>
<td>LESENSE_SYNCBUSY</td>
<td>R</td>
<td>Synchronization Busy Register</td>
</tr>
<tr>
<td>0x054</td>
<td>LESENSE_ROUTE</td>
<td>RW</td>
<td>I/O Routing Register</td>
</tr>
<tr>
<td>0x058</td>
<td>LESENSE_POWERDOWN</td>
<td>RW</td>
<td>LESENSE RAM power-down register</td>
</tr>
<tr>
<td>0x200</td>
<td>LESENSE_ST0_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x204</td>
<td>LESENSE_ST0_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x208</td>
<td>LESENSE_ST1_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x20C</td>
<td>LESENSE_ST1_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x210</td>
<td>LESENSE_ST2_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x214</td>
<td>LESENSE_ST2_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x218</td>
<td>LESENSE_ST3_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x21C</td>
<td>LESENSE_ST3_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x220</td>
<td>LESENSE_ST4_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x224</td>
<td>LESENSE_ST4_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x228</td>
<td>LESENSE_ST5_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x22C</td>
<td>LESENSE_ST5_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x230</td>
<td>LESENSE_ST6_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x234</td>
<td>LESENSE_ST6_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x238</td>
<td>LESENSE_ST7_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x23C</td>
<td>LESENSE_ST7_TCONFB</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x240</td>
<td>LESENSE_ST8_TCONFA</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>Offset</td>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
<td>------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>0x244</td>
<td>LESENSE_ST8_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x248</td>
<td>LESENSE_ST9_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x24C</td>
<td>LESENSE_ST9_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x250</td>
<td>LESENSE_ST10_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x254</td>
<td>LESENSE_ST10_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x258</td>
<td>LESENSE_ST11_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x25C</td>
<td>LESENSE_ST11_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x260</td>
<td>LESENSE_ST12_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x264</td>
<td>LESENSE_ST12_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x268</td>
<td>LESENSE_ST13_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x26C</td>
<td>LESENSE_ST13_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x270</td>
<td>LESENSE_ST14_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x274</td>
<td>LESENSE_ST14_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x278</td>
<td>LESENSE_ST15_TCONF</td>
<td>RW</td>
<td>State transition configuration A</td>
</tr>
<tr>
<td>0x27C</td>
<td>LESENSE_ST15_TCONF</td>
<td>RW</td>
<td>State transition configuration B</td>
</tr>
<tr>
<td>0x280</td>
<td>LESENSE_BUF0_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x284</td>
<td>LESENSE_BUF1_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x288</td>
<td>LESENSE_BUF2_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x28C</td>
<td>LESENSE_BUF3_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x290</td>
<td>LESENSE_BUF4_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x294</td>
<td>LESENSE_BUF5_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x298</td>
<td>LESENSE_BUF6_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x29C</td>
<td>LESENSE_BUF7_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2A0</td>
<td>LESENSE_BUF8_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2A4</td>
<td>LESENSE_BUF9_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2A8</td>
<td>LESENSE_BUF10_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2AC</td>
<td>LESENSE_BUF11_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2B0</td>
<td>LESENSE_BUF12_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2B4</td>
<td>LESENSE_BUF13_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2B8</td>
<td>LESENSE_BUF14_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2BC</td>
<td>LESENSE_BUF15_DATA</td>
<td>RW</td>
<td>Scan results</td>
</tr>
<tr>
<td>0x2C0</td>
<td>LESENSE_CH0_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2C4</td>
<td>LESENSE_CH0_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2C8</td>
<td>LESENSE_CH0_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2D0</td>
<td>LESENSE_CH1_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2D4</td>
<td>LESENSE_CH1_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2D8</td>
<td>LESENSE_CH1_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2E0</td>
<td>LESENSE_CH2_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2E4</td>
<td>LESENSE_CH2_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2E8</td>
<td>LESENSE_CH2_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2F0</td>
<td>LESENSE_CH3_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x2F4</td>
<td>LESENSE_CH3_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>Offset</td>
<td>Name</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------</td>
<td>------</td>
<td>-------------------</td>
</tr>
<tr>
<td>0x2F8</td>
<td>LESENSE_CH3_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x300</td>
<td>LESENSE_CH4_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x304</td>
<td>LESENSE_CH4_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x308</td>
<td>LESENSE_CH4_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x310</td>
<td>LESENSE_CH5_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x314</td>
<td>LESENSE_CH5_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x318</td>
<td>LESENSE_CH5_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x320</td>
<td>LESENSE_CH6_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x324</td>
<td>LESENSE_CH6_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x328</td>
<td>LESENSE_CH6_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x330</td>
<td>LESENSE_CH7_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x334</td>
<td>LESENSE_CH7_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x338</td>
<td>LESENSE_CH7_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x340</td>
<td>LESENSE_CH8_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x344</td>
<td>LESENSE_CH8_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x348</td>
<td>LESENSE_CH8_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x350</td>
<td>LESENSE_CH9_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x354</td>
<td>LESENSE_CH9_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x358</td>
<td>LESENSE_CH9_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x360</td>
<td>LESENSE_CH10_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x364</td>
<td>LESENSE_CH10_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x368</td>
<td>LESENSE_CH10_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x370</td>
<td>LESENSE_CH11_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x374</td>
<td>LESENSE_CH11_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x378</td>
<td>LESENSE_CH11_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x380</td>
<td>LESENSE_CH12_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x384</td>
<td>LESENSE_CH12_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x388</td>
<td>LESENSE_CH12_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x390</td>
<td>LESENSE_CH13_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x394</td>
<td>LESENSE_CH13_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x398</td>
<td>LESENSE_CH13_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3A0</td>
<td>LESENSE_CH14_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3A4</td>
<td>LESENSE_CH14_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3A8</td>
<td>LESENSE_CH14_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3B0</td>
<td>LESENSE_CH15_TIMING</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3B4</td>
<td>LESENSE_CH15_INTERACT</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
<tr>
<td>0x3B8</td>
<td>LESENSE_CH15_EVAL</td>
<td>RW</td>
<td>Scan configuration</td>
</tr>
</tbody>
</table>

### 21.5 Register Description

#### 21.5.1 LESENSE_CTRL - Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18) .
## Offset 0x000

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:23</td>
<td>31:23</td>
<td>RESERVED</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>DEBUGRUN</td>
<td>0</td>
<td>RW</td>
<td>Debug Mode Run Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Set to keep LESENSE running in debug mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: LESENSE can not start new scans in debug mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: LESENSE can start new scans in debug mode</td>
</tr>
<tr>
<td>21:20</td>
<td>21:20</td>
<td>DMAWU</td>
<td>0x0</td>
<td>RW</td>
<td>DMA wakeup from EM2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Mode Description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: DISABLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>No DMA wakeup from EM2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: BUFDATAV</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DMA wakeup from EM2 when data is valid in the result buffer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2: BUFLEVEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DMA wakeup from EM2 when the result buffer is full/halffull depending on BUFIDL configuration</td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td>RESERVED</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td>BUFIDL</td>
<td>0</td>
<td>RW</td>
<td>Result buffer interrupt and DMA trigger level</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Mode Description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: HALFFULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DMA and interrupt flags set when result buffer is halffull</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: FULL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>DMA and interrupt flags set when result buffer is full</td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td>STRSCANRES</td>
<td>0</td>
<td>RW</td>
<td>Enable storing of SCANRES</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When set, SCANRES will be stored in the result buffer after each scan</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>BUFOW</td>
<td>0</td>
<td>RW</td>
<td>Result buffer overwrite</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>If set, LESENSE will always write to the result buffer, even if it is full</td>
</tr>
<tr>
<td>15:14</td>
<td>15:14</td>
<td>RESERVED</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>DUALSAMPLE</td>
<td>0</td>
<td>RW</td>
<td>Enable dual sample mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>When set, both ACMPs will be sampled simultaneously.</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>RESERVED</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>ALTEXMAP</td>
<td>0</td>
<td>RW</td>
<td>Alternative excitation map</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Value Mode Description</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0: ALTEX</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Alternative excitation is mapped to the LES_ALTEX pins.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1: ACMP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Alternative excitation is mapped to the pins of the other ACMP.</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>ACMP1INV</td>
<td>0</td>
<td>RW</td>
<td>Invert analog comparator 1 output</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>ACMP0INV</td>
<td>0</td>
<td>RW</td>
<td>Invert analog comparator 0 output</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>RESERVED</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>Bit</td>
<td>Name</td>
<td>Reset</td>
<td>Access</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----------------</td>
<td>-------</td>
<td>--------</td>
<td>-------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>7:6</td>
<td>SCANCONF</td>
<td>0x0</td>
<td>RW</td>
<td>Select scan configuration</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>These bits control which CHx_CONF registers to be used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>DIRMAP</td>
<td></td>
<td>The channel configuration register registers used are directly mapped to the channel number.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>INVMAP</td>
<td></td>
<td>The channel configuration register registers used are CHx.CONF for channels 0-7 and CHx.8.CONF for channels 8-15.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>TOGGLE</td>
<td></td>
<td>The channel configuration register registers used toggles between CHx.CONF and CHx.8.CONF when channel x triggers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>DECDEF</td>
<td></td>
<td>The decoder state defines the CONF registers to be used.</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).</td>
<td></td>
</tr>
<tr>
<td>4:2</td>
<td>PRSSEL</td>
<td>0x0</td>
<td>RW</td>
<td>Scan start PRS select</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Select PRS source for scan start if SCANMODE is set to PRS.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>PRSCH0</td>
<td></td>
<td>PRS Channel 0 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>PRSCH1</td>
<td></td>
<td>PRS Channel 1 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>PRSCH2</td>
<td></td>
<td>PRS Channel 2 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>PRSCH3</td>
<td></td>
<td>PRS Channel 3 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>PRSCH4</td>
<td></td>
<td>PRS Channel 4 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>PRSCH5</td>
<td></td>
<td>PRS Channel 5 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>PRSCH6</td>
<td></td>
<td>PRS Channel 6 selected as input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>PRSCH7</td>
<td></td>
<td>PRS Channel 7 selected as input</td>
<td></td>
</tr>
<tr>
<td>1:0</td>
<td>SCANMODE</td>
<td>0x0</td>
<td>RW</td>
<td>Configure scan mode</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>These bits control how the scan frequency is decided</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>PERIODIC</td>
<td></td>
<td>A new scan is started each time the period counter overflows</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>ONESHOT</td>
<td></td>
<td>A single scan is performed when START in CMD is set</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>PRS</td>
<td></td>
<td>Pulse on PRS channel</td>
<td></td>
</tr>
</tbody>
</table>

### 21.5.2 LESENSE_TIMCTRL - Timing Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x004</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>29</td>
<td>28</td>
</tr>
<tr>
<td>27</td>
<td>26</td>
</tr>
<tr>
<td>25</td>
<td>24</td>
</tr>
<tr>
<td>23</td>
<td>22</td>
</tr>
<tr>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>19</td>
<td>18</td>
</tr>
<tr>
<td>17</td>
<td>16</td>
</tr>
<tr>
<td>15</td>
<td>14</td>
</tr>
<tr>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:24</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).</td>
</tr>
<tr>
<td>23:22</td>
<td>STARTDLY</td>
<td>0x0</td>
<td>RW</td>
<td>Start delay configuration</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Delay sensor interaction STARTDELAY LFACLK LESENSE cycles for each channel</td>
</tr>
<tr>
<td>21:20</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).</td>
</tr>
<tr>
<td>19:12</td>
<td>PCTOP</td>
<td>0x000</td>
<td>RW</td>
<td>Period counter top value</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>These bits contain the top value for the period counter.</td>
</tr>
<tr>
<td>11</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).</td>
</tr>
</tbody>
</table>
### Bit Name Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:8</td>
<td>PCPRESC</td>
<td>0x0</td>
<td>RW</td>
<td>Period counter prescaling</td>
</tr>
</tbody>
</table>

#### Value | Mode | Description

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DIV1</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>1</td>
<td>DIV2</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>2</td>
<td>DIV4</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;4&lt;/sup&gt;</td>
</tr>
<tr>
<td>3</td>
<td>DIV8</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;8&lt;/sup&gt;</td>
</tr>
<tr>
<td>4</td>
<td>DIV16</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;16&lt;/sup&gt;</td>
</tr>
<tr>
<td>5</td>
<td>DIV32</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;32&lt;/sup&gt;</td>
</tr>
<tr>
<td>6</td>
<td>DIV64</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;64&lt;/sup&gt;</td>
</tr>
<tr>
<td>7</td>
<td>DIV128</td>
<td>The period counter clock frequency is LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;128&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DIV1</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;1&lt;/sup&gt;</td>
</tr>
<tr>
<td>1</td>
<td>DIV2</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;2&lt;/sup&gt;</td>
</tr>
<tr>
<td>2</td>
<td>DIV4</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;4&lt;/sup&gt;</td>
</tr>
<tr>
<td>3</td>
<td>DIV8</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;8&lt;/sup&gt;</td>
</tr>
<tr>
<td>4</td>
<td>DIV16</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;16&lt;/sup&gt;</td>
</tr>
<tr>
<td>5</td>
<td>DIV32</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;32&lt;/sup&gt;</td>
</tr>
<tr>
<td>6</td>
<td>DIV64</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;64&lt;/sup&gt;</td>
</tr>
<tr>
<td>7</td>
<td>DIV128</td>
<td>Low frequency timer is clocked with LFACLK&lt;sub&gt;LESENSE&lt;/sub&gt;&lt;sup&gt;128&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DIV1</td>
<td>High frequency timer is clocked with AUXHFRCO/1</td>
</tr>
<tr>
<td>1</td>
<td>DIV2</td>
<td>High frequency timer is clocked with AUXHFRCO/2</td>
</tr>
<tr>
<td>2</td>
<td>DIV4</td>
<td>High frequency timer is clocked with AUXHFRCO/4</td>
</tr>
<tr>
<td>3</td>
<td>DIV8</td>
<td>High frequency timer is clocked with AUXHFRCO/8</td>
</tr>
</tbody>
</table>

### 21.5.3 LESENSE_PERCTRL - Peripheral Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).
<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>25:24</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>23:22</td>
<td>ACMP1MODE</td>
<td>0x0</td>
<td>RW</td>
<td>ACMP1 mode</td>
</tr>
<tr>
<td></td>
<td>Configure how LESENSE controls ACMP1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>NORMAL</td>
<td></td>
<td>The analog comparators and DAC are shut down when LESENSE is idle</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>KEEPACMPWARM</td>
<td></td>
<td>The analog comparators are kept powered up when LESENSE is idle</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>KEEPDACWARM</td>
<td></td>
<td>The DAC is kept powered up when LESENSE is idle</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>KEEPACMPDACWARM</td>
<td></td>
<td>The analog comparators and DAC are kept powered up when LESENSE is idle</td>
<td></td>
</tr>
<tr>
<td>21:20</td>
<td>ACMP0MODE</td>
<td>0x0</td>
<td>RW</td>
<td>ACMP0 mode</td>
</tr>
<tr>
<td></td>
<td>Configure how LESENSE controls ACMP0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
<td>LESENSE does not control ACMP1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>MUX</td>
<td></td>
<td>LESENSE controls the input mux (POSSEL) of ACMP1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MUXTHRES</td>
<td></td>
<td>LESENSE controls the input mux and the threshold value (VDDLEVEL) of ACMP0</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>18</td>
<td>DACREF</td>
<td>0</td>
<td>RW</td>
<td>DAC bandgap reference used</td>
</tr>
<tr>
<td></td>
<td>Set to BANDGAP if the DAC is configured to use bandgap reference</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>VDD</td>
<td></td>
<td>DAC uses VDD reference</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>BANDGAP</td>
<td></td>
<td>DAC uses bandgap reference</td>
<td></td>
</tr>
<tr>
<td>17:15</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>14:10</td>
<td>DACPRESCE</td>
<td>0x00</td>
<td>RW</td>
<td>DAC prescaler configuration.</td>
</tr>
<tr>
<td></td>
<td>Prescaling factor of DACPRESCE+1 for the LESENSE DAC interface</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:8</td>
<td>DACCH1OUT</td>
<td>0x0</td>
<td>RW</td>
<td>DAC channel 1 output mode</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
<td>DAC CH1 output to pin and ACMP/ADC disabled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>PIN</td>
<td></td>
<td>DAC CH1 output to pin enabled, output to ADC and ACMP disabled</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ADCACMP</td>
<td></td>
<td>DAC CH1 output to pin disabled, output to ADC and ACMP enabled</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PINADCACMP</td>
<td></td>
<td>DAC CH1 output to pin, ADC, and ACMP enabled.</td>
<td></td>
</tr>
<tr>
<td>7:6</td>
<td>DACCH0OUT</td>
<td>0x0</td>
<td>RW</td>
<td>DAC channel 0 output mode</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
<td>DAC CH0 output to pin and ACMP/ADC disabled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>PIN</td>
<td></td>
<td>DAC CH0 output to pin enabled, output to ADC and ACMP disabled</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ADCACMP</td>
<td></td>
<td>DAC CH0 output to pin disabled, output to ADC and ACMP enabled</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PINADCACMP</td>
<td></td>
<td>DAC CH0 output to pin, ADC, and ACMP enabled.</td>
<td></td>
</tr>
<tr>
<td>5:4</td>
<td>DACCH1CONV</td>
<td>0x0</td>
<td>RW</td>
<td>DAC channel 1 conversion mode</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
<td>LESENSE does not control DAC CH1.</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>CONTINUOUS</td>
<td></td>
<td>DAC channel 1 is driven in continuous mode.</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>SAMPLEHOLD</td>
<td></td>
<td>DAC channel 1 is driven in sample hold mode.</td>
<td></td>
</tr>
</tbody>
</table>

...the world's most energy friendly microcontrollers
### Bit Name Reset Access Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>SAMPLEOFF</td>
<td>0x0</td>
<td>RW</td>
<td>DAC channel 1 is driven in sample off mode.</td>
</tr>
</tbody>
</table>

#### 3:2 DACCH0CONV

- **Value**: 0x0
- **Mode**: RW
- **Description**: DAC channel 0 conversion mode

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>LESENSE does not control DAC CH0.</td>
</tr>
<tr>
<td>1</td>
<td>CONTINUOUS</td>
<td>DAC channel 0 is driven in continuous mode.</td>
</tr>
<tr>
<td>2</td>
<td>SAMPLEHOLD</td>
<td>DAC channel 0 is driven in sample hold mode.</td>
</tr>
<tr>
<td>3</td>
<td>SAMPLEOFF</td>
<td>DAC channel 0 is driven in sample off mode.</td>
</tr>
</tbody>
</table>

#### 1 DACCH1DATA

Configure DAC data control.

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DACDATA</td>
<td>DAC data is defined by CH1DATA in the DAC interface.</td>
</tr>
<tr>
<td>1</td>
<td>ACMPTHRES</td>
<td>DAC data is defined by ACMPTHRES in CHx_INTERACT.</td>
</tr>
</tbody>
</table>

#### 0 DACCH0DATA

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DACDATA</td>
<td>DAC data is defined by CH0DATA in the DAC interface.</td>
</tr>
<tr>
<td>1</td>
<td>ACMPTHRES</td>
<td>DAC data is defined by ACMPTHRES in CHx_INTERACT.</td>
</tr>
</tbody>
</table>

### 21.5.4 LESENSE_DECCTRL - Decoder control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

#### Offset Bit Position

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00C</td>
<td></td>
</tr>
</tbody>
</table>

#### Bit Name Reset Access Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:25</td>
<td>RESERVED</td>
<td>0x0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>24:22</td>
<td>PRSSEL3</td>
<td>0x0</td>
<td>RW</td>
<td>Select PRS input for bit 3 of the LESENSE decoder</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PRSCH0</td>
<td>PRS Channel 0 selected as input</td>
</tr>
<tr>
<td>1</td>
<td>PRSCH1</td>
<td>PRS Channel 1 selected as input</td>
</tr>
<tr>
<td>2</td>
<td>PRSCH2</td>
<td>PRS Channel 2 selected as input</td>
</tr>
<tr>
<td>3</td>
<td>PRSCH3</td>
<td>PRS Channel 3 selected as input</td>
</tr>
<tr>
<td>4</td>
<td>PRSCH4</td>
<td>PRS Channel 4 selected as input</td>
</tr>
<tr>
<td>5</td>
<td>PRSCH5</td>
<td>PRS Channel 5 selected as input</td>
</tr>
<tr>
<td>6</td>
<td>PRSCH6</td>
<td>PRS Channel 6 selected as input</td>
</tr>
<tr>
<td>7</td>
<td>PRSCH7</td>
<td>PRS Channel 7 selected as input</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20:18</td>
<td>PRSSEL2</td>
<td>0x0</td>
<td>RW</td>
<td>Select PRS input for bit 2 of the LESENSE decoder</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PRSCH0</td>
<td>PRS Channel 0 selected as input</td>
</tr>
<tr>
<td>1</td>
<td>PRSCH1</td>
<td>PRS Channel 1 selected as input</td>
</tr>
<tr>
<td>2</td>
<td>PRSCH2</td>
<td>PRS Channel 2 selected as input</td>
</tr>
<tr>
<td>3</td>
<td>PRSCH3</td>
<td>PRS Channel 3 selected as input</td>
</tr>
<tr>
<td>4</td>
<td>PRSCH4</td>
<td>PRS Channel 4 selected as input</td>
</tr>
<tr>
<td>5</td>
<td>PRSCH5</td>
<td>PRS Channel 5 selected as input</td>
</tr>
<tr>
<td>6</td>
<td>PRSCH6</td>
<td>PRS Channel 6 selected as input</td>
</tr>
<tr>
<td>7</td>
<td>PRSCH7</td>
<td>PRS Channel 7 selected as input</td>
</tr>
<tr>
<td>Bit</td>
<td>Name</td>
<td>Reset</td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
</tr>
<tr>
<td>0</td>
<td>PRSCH0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>PRSCH1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PRSCH2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>PRSCH3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PRSCH4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>PRSCH5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>PRSCH6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>PRSCH7</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>17</th>
<th>Reserved</th>
<th>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</th>
</tr>
</thead>
</table>

**16:14 PRSSEL1** 0x0 RW

Select PRS input for the bit 1 of the LESENSE decoder

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>PRSCH0</td>
<td></td>
<td></td>
<td>PRS Channel 0 selected as input</td>
</tr>
<tr>
<td>1</td>
<td>PRSCH1</td>
<td></td>
<td></td>
<td>PRS Channel 1 selected as input</td>
</tr>
<tr>
<td>2</td>
<td>PRSCH2</td>
<td></td>
<td></td>
<td>PRS Channel 2 selected as input</td>
</tr>
<tr>
<td>3</td>
<td>PRSCH3</td>
<td></td>
<td></td>
<td>PRS Channel 3 selected as input</td>
</tr>
<tr>
<td>4</td>
<td>PRSCH4</td>
<td></td>
<td></td>
<td>PRS Channel 4 selected as input</td>
</tr>
<tr>
<td>5</td>
<td>PRSCH5</td>
<td></td>
<td></td>
<td>PRS Channel 5 selected as input</td>
</tr>
<tr>
<td>6</td>
<td>PRSCH6</td>
<td></td>
<td></td>
<td>PRS Channel 6 selected as input</td>
</tr>
<tr>
<td>7</td>
<td>PRSCH7</td>
<td></td>
<td></td>
<td>PRS Channel 7 selected as input</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13</th>
<th>Reserved</th>
<th>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</th>
</tr>
</thead>
</table>

**12:10 PRSSEL0** 0x0 RW

Select PRS input for the bit 0 of the LESENSE decoder

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>PRSCH0</td>
<td></td>
<td></td>
<td>PRS Channel 0 selected as input</td>
</tr>
<tr>
<td>1</td>
<td>PRSCH1</td>
<td></td>
<td></td>
<td>PRS Channel 1 selected as input</td>
</tr>
<tr>
<td>2</td>
<td>PRSCH2</td>
<td></td>
<td></td>
<td>PRS Channel 2 selected as input</td>
</tr>
<tr>
<td>3</td>
<td>PRSCH3</td>
<td></td>
<td></td>
<td>PRS Channel 3 selected as input</td>
</tr>
<tr>
<td>4</td>
<td>PRSCH4</td>
<td></td>
<td></td>
<td>PRS Channel 4 selected as input</td>
</tr>
<tr>
<td>5</td>
<td>PRSCH5</td>
<td></td>
<td></td>
<td>PRS Channel 5 selected as input</td>
</tr>
<tr>
<td>6</td>
<td>PRSCH6</td>
<td></td>
<td></td>
<td>PRS Channel 6 selected as input</td>
</tr>
<tr>
<td>7</td>
<td>PRSCH7</td>
<td></td>
<td></td>
<td>PRS Channel 7 selected as input</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9</th>
<th>Reserved</th>
<th>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</th>
</tr>
</thead>
</table>

**8** INPUT 0 RW

Select input to the LESENSE decoder

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>SENSORSTATE</td>
<td></td>
<td></td>
<td>The SENSORSTATE register is used as input to the decoder.</td>
</tr>
<tr>
<td>1</td>
<td>PRS</td>
<td></td>
<td></td>
<td>PRS channels are used as input to the decoder.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7</th>
<th>PRSCNT</th>
<th>0</th>
<th>RW</th>
<th>Enable count mode on decoder PRS channels 0 and 1</th>
</tr>
</thead>
</table>

When set, decoder PRS0 and PRS1 will be used to produce output which can be used by a PCNT to count up or down.

<table>
<thead>
<tr>
<th>6</th>
<th>HYSTIRQ</th>
<th>0</th>
<th>RW</th>
<th>Enable decoder hysteresis on interrupt requests</th>
</tr>
</thead>
</table>

When set, hysteresis is enabled in the decoder, suppressing interrupt requests.

<table>
<thead>
<tr>
<th>5</th>
<th>HYSTPRS2</th>
<th>0</th>
<th>RW</th>
<th>Enable decoder hysteresis on PRS2 output</th>
</tr>
</thead>
</table>

When set, hysteresis is enabled in the decoder, suppressing changes on PRS channel 2.

<table>
<thead>
<tr>
<th>4</th>
<th>HYSTPRS1</th>
<th>0</th>
<th>RW</th>
<th>Enable decoder hysteresis on PRS1 output</th>
</tr>
</thead>
</table>

When set, hysteresis is enabled in the decoder, suppressing changes on PRS channel 1.

<table>
<thead>
<tr>
<th>3</th>
<th>HYSTPRS0</th>
<th>0</th>
<th>RW</th>
<th>Enable decoder hysteresis on PRS0 output</th>
</tr>
</thead>
</table>

When set, hysteresis is enabled in the decoder, suppressing changes on PRS channel 0.
<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>INTMAP</td>
<td>0</td>
<td>RW</td>
<td>Enable decoder to channel interrupt mapping</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(When set, a transition from state x in the decoder will set interrupt flag CHx)</td>
</tr>
<tr>
<td>1</td>
<td>ERRCHK</td>
<td>0</td>
<td>RW</td>
<td>Enable check of current state</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(When set, the decoder checks the current state in addition to the states defined in TCONF)</td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>0</td>
<td>RW</td>
<td>Disable the decoder</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>(When set, the decoder is disabled. When disabled the decoder will keep its current state)</td>
</tr>
</tbody>
</table>

21.5.5 LESENSE_BIASCTRL - Bias Control Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x010</td>
<td>31:0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>29</td>
<td></td>
</tr>
<tr>
<td></td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>26</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td></td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reset</th>
<th>Access</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RW</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:2</td>
<td>Reserved</td>
<td>0x0</td>
<td>RW</td>
<td>Select bias mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DUTYCYCLE</td>
<td>Bias module duty cycled between low power and high accuracy mode</td>
</tr>
<tr>
<td>1</td>
<td>HIGHACC</td>
<td>Bias module always in high accuracy mode</td>
</tr>
<tr>
<td>2</td>
<td>DONTTOUCH</td>
<td>Bias module is controlled by the EMU and not affected by LESENSE</td>
</tr>
</tbody>
</table>

21.5.6 LESENSE_CMD - Command Register

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x014</td>
<td>31:0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>29</td>
<td></td>
</tr>
<tr>
<td></td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>26</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td></td>
</tr>
<tr>
<td></td>
<td>24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>21</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>18</td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reset</th>
<th>Access</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>W1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:4</td>
<td>Reserved</td>
<td>0</td>
<td>W1</td>
<td>Clear result buffer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>CLEARBUF</td>
<td>Bias module is controlled by the EMU and not affected by LESENSE</td>
</tr>
</tbody>
</table>

2 | DECODE  | 0     | W1     | Start decoder |

1 | STOP    | 0     | W1     | Stop scanning of sensors |
## 21.5.7 LESENSE_CHEN - Channel enable Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x018</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**
- 0x0000

**Access**
- RW

**Name**
- CHEN

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:16</td>
<td>Reserved</td>
<td>0x0000</td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>15:0</td>
<td>CHEN</td>
<td>0x0000</td>
<td>RW</td>
<td>Enable scan channel Set bit X to enable channel X</td>
</tr>
</tbody>
</table>

## 21.5.8 LESENSE_SCANRES - Scan result register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x01C</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**
- 0x0000

**Access**
- R

**Name**
- SCANRES

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:16</td>
<td>Reserved</td>
<td>0x0000</td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>15:0</td>
<td>SCANRES</td>
<td>0x0000</td>
<td>R</td>
<td>Scan results Bit X will be set depending on channel X evaluation</td>
</tr>
</tbody>
</table>

## 21.5.9 LESENSE_STATUS - Status Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).
The world's most energy friendly microcontrollers

Offset | Bit Position |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x020</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

Reset | Access |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>DACACTIVE</td>
<td>SCANACTIVE</td>
</tr>
<tr>
<td>RUNNING</td>
<td>BUFFULL</td>
</tr>
<tr>
<td>BUFHALFFULL</td>
<td>BUFDATAV</td>
</tr>
</tbody>
</table>

**Bit** | **Name** | **Reset** | **Access** | **Description**
---|---|---|---|---
31:6 | Reserved | 0 | R | To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)
5 | DACACTIVE | 0 | R | LESENSE DAC interface is active
4 | SCANACTIVE | 0 | R | LESENSE is currently interfacing sensors.
3 | RUNNING | 0 | R | LESENSE is active
2 | BUFFULL | 0 | R | Result buffer full
Set when the result buffer is full
1 | BUFHALFFULL | 0 | R | Result buffer half full
Set when the result buffer is half full
0 | BUFDATAV | 0 | R | Result data valid
Set when data is available in the result buffer. Cleared when the buffer is empty.

**21.5.10 LESENSE_PTR - Result buffer pointers (Async Reg)**

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

Offset | Bit Position |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0x024</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

Reset | Access |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>WR</td>
<td>RD</td>
</tr>
</tbody>
</table>

**Bit** | **Name** | **Reset** | **Access** | **Description**
---|---|---|---|---
31:9 | Reserved | 0x0 | R | To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)
8:5 | WR | 0x0 | R | Result buffer write pointer.
These bits show the next index in the result buffer to be written to. Incremented when LESENSE writes to result buffer
4 | Reserved | 0x0 | R | To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)
3:0 | RD | 0x0 | R | Result buffer read pointer.
These bits show the index of the oldest unread data in the result buffer. Incremented on read from BUFDATA.

**21.5.11 LESENSE_BUFDATA - Result buffer data register (Async Reg)**

For more information about Asynchronous Registers please see Section 5.3 (p. 18).
## 21.5.12 LESENSE_CURCH - Current channel index (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x02C</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**

<table>
<thead>
<tr>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:16</td>
<td>Reserved</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
<td></td>
</tr>
<tr>
<td>15:0</td>
<td>BUFDATA</td>
<td>0xFFFF</td>
<td>R</td>
</tr>
</tbody>
</table>

This register can be used to read the oldest unread data from the result buffer.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:16</td>
<td>Reserved</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:0</td>
<td>CURCH</td>
<td>0x0</td>
<td>R</td>
<td>Shows the index of the current channel</td>
</tr>
</tbody>
</table>

## 21.5.13 LESENSE_DECSTATE - Current decoder state (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x030</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**

<table>
<thead>
<tr>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:4</td>
<td>Reserved</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
<td></td>
</tr>
<tr>
<td>3:0</td>
<td>DECSTATE</td>
<td>0x0</td>
<td>RWH</td>
</tr>
</tbody>
</table>
21.5.14 LESENSE_SENSORSTATE - Decoder input register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x034</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:4</td>
<td>Reserved</td>
<td>0x0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| 3:0   | SENSORSTATE | 0x0   | RWH    | Shows the status of sensors chosen as input to the decoder |

21.5.15 LESENSE_IDLECONF - GPIO Idlephase configuration (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x038</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:30</td>
<td>CH15</td>
<td>0x0</td>
<td>RW</td>
<td>Channel 15 idlephase configuration</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>CH15 output is disabled in idle phase</td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>CH15 output is high in idle phase</td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>CH15 output is low in idle phase</td>
</tr>
<tr>
<td>3</td>
<td>DACCH1</td>
<td>CH15 output is connected to DAC CH1 output in idle phase</td>
</tr>
</tbody>
</table>

| 29:28 | CH14      | 0x0   | RW     | Channel 14 idlephase configuration |

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>CH14 output is disabled in idle phase</td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>CH14 output is high in idle phase</td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>CH14 output is low in idle phase</td>
</tr>
<tr>
<td>3</td>
<td>DACCH1</td>
<td>CH14 output is connected to DAC CH1 output in idle phase</td>
</tr>
</tbody>
</table>

| 27:26 | CH13      | 0x0   | RW     | Channel 13 idlephase configuration |

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>CH13 output is disabled in idle phase</td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>CH13 output is high in idle phase</td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>CH13 output is low in idle phase</td>
</tr>
<tr>
<td>3</td>
<td>DACCH1</td>
<td>CH13 output is connected to DAC CH1 output in idle phase</td>
</tr>
<tr>
<td>Bit</td>
<td>Name</td>
<td>Reset</td>
</tr>
<tr>
<td>------</td>
<td>----------</td>
<td>-------</td>
</tr>
<tr>
<td>25:24</td>
<td>CH12</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>DACCH1</td>
<td></td>
</tr>
<tr>
<td>23:22</td>
<td>CH11</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>21:20</td>
<td>CH10</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>19:18</td>
<td>CH9</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>17:16</td>
<td>CH8</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>15:14</td>
<td>CH7</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>13:12</td>
<td>CH6</td>
<td>0x0</td>
</tr>
<tr>
<td>Value</td>
<td>Mode</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td></td>
</tr>
<tr>
<td>Bit</td>
<td>Name</td>
<td>Reset</td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>-------</td>
</tr>
<tr>
<td>11:10</td>
<td>CH5</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:8</td>
<td>CH4</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7:6</td>
<td>CH3</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5:4</td>
<td>CH2</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:2</td>
<td>CH1</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1:0</td>
<td>CH0</td>
<td>0x0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

21.5.16 LESENSE_ALTEXCONF - Alternative excite pin configuration (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).
### Bit Offset Table

**Offset**: 0x03C

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:24</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>23</td>
<td>AEX7</td>
<td>0</td>
<td>RW</td>
<td>ALTEX7 always excite enable</td>
</tr>
<tr>
<td>22</td>
<td>AEX6</td>
<td>0</td>
<td>RW</td>
<td>ALTEX6 always excite enable</td>
</tr>
<tr>
<td>21</td>
<td>AEX5</td>
<td>0</td>
<td>RW</td>
<td>ALTEX5 always excite enable</td>
</tr>
<tr>
<td>20</td>
<td>AEX4</td>
<td>0</td>
<td>RW</td>
<td>ALTEX4 always excite enable</td>
</tr>
<tr>
<td>19</td>
<td>AEX3</td>
<td>0</td>
<td>RW</td>
<td>ALTEX3 always excite enable</td>
</tr>
<tr>
<td>18</td>
<td>AEX2</td>
<td>0</td>
<td>RW</td>
<td>ALTEX2 always excite enable</td>
</tr>
<tr>
<td>17</td>
<td>AEX1</td>
<td>0</td>
<td>RW</td>
<td>ALTEX1 always excite enable</td>
</tr>
<tr>
<td>16</td>
<td>AEX0</td>
<td>0</td>
<td>RW</td>
<td>ALTEX0 always excite enable</td>
</tr>
<tr>
<td>15:14</td>
<td>IDLECONF7</td>
<td>0x0</td>
<td>RW</td>
<td>ALTEX7 idle phase configuration</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX7 output is disabled in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>ALTEX7 output is high in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>ALTEX7 output is low in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13:12</td>
<td>IDLECONF6</td>
<td>0x0</td>
<td>RW</td>
<td>ALTEX6 idle phase configuration</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX6 output is disabled in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>ALTEX6 output is high in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>ALTEX6 output is low in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11:10</td>
<td>IDLECONF5</td>
<td>0x0</td>
<td>RW</td>
<td>ALTEX5 idle phase configuration</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX5 output is disabled in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>ALTEX5 output is high in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>ALTEX5 output is low in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9:8</td>
<td>IDLECONF4</td>
<td>0x0</td>
<td>RW</td>
<td>ALTEX4 idle phase configuration</td>
</tr>
<tr>
<td></td>
<td>Value</td>
<td>Mode</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX4 output is disabled in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>ALTEX4 output is high in idle phase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>ALTEX4 output is low in idle phase</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Bit Value \( \times \) Mode \( \times \) Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>7:6</td>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX4 output is disabled in idle phase</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>HIGH</td>
<td>ALTEX4 output is high in idle phase</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>LOW</td>
<td>ALTEX4 output is low in idle phase</td>
</tr>
</tbody>
</table>

### 7:6 IDLECONF3 0x0 RW ALTEX3 idle phase configuration

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5:4</td>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX3 output is disabled in idle phase</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>HIGH</td>
<td>ALTEX3 output is high in idle phase</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>LOW</td>
<td>ALTEX3 output is low in idle phase</td>
</tr>
</tbody>
</table>

### 5:4 IDLECONF2 0x0 RW ALTEX2 idle phase configuration

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3:2</td>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX2 output is disabled in idle phase</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>HIGH</td>
<td>ALTEX2 output is high in idle phase</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>LOW</td>
<td>ALTEX2 output is low in idle phase</td>
</tr>
</tbody>
</table>

### 3:2 IDLECONF1 0x0 RW ALTEX1 idle phase configuration

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1:0</td>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX1 output is disabled in idle phase</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>HIGH</td>
<td>ALTEX1 output is high in idle phase</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>LOW</td>
<td>ALTEX1 output is low in idle phase</td>
</tr>
</tbody>
</table>

### 1:0 IDLECONF0 0x0 RW ALTEX0 idle phase configuration

<table>
<thead>
<tr>
<th>Bit</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>DISABLE</td>
<td>ALTEX0 output is disabled in idle phase</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>HIGH</td>
<td>ALTEX0 output is high in idle phase</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>LOW</td>
<td>ALTEX0 output is low in idle phase</td>
</tr>
</tbody>
</table>

### 21.5.17 LESENSE_IF - Interrupt Flag Register

#### Offset Bit Position

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x040</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

#### Reset

<table>
<thead>
<tr>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNTOF</td>
<td>0</td>
<td>R</td>
</tr>
<tr>
<td>BUFOF</td>
<td>0</td>
<td>R</td>
</tr>
</tbody>
</table>

#### Access

<table>
<thead>
<tr>
<th>Name</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNTOF</td>
<td>R</td>
</tr>
<tr>
<td>BUFOF</td>
<td>R</td>
</tr>
<tr>
<td>CH0</td>
<td>R</td>
</tr>
<tr>
<td>CH1</td>
<td>R</td>
</tr>
<tr>
<td>CH2</td>
<td>R</td>
</tr>
<tr>
<td>CH3</td>
<td>R</td>
</tr>
<tr>
<td>CH4</td>
<td>R</td>
</tr>
<tr>
<td>CH5</td>
<td>R</td>
</tr>
<tr>
<td>CH6</td>
<td>R</td>
</tr>
<tr>
<td>CH7</td>
<td>R</td>
</tr>
<tr>
<td>CH8</td>
<td>R</td>
</tr>
<tr>
<td>CH9</td>
<td>R</td>
</tr>
<tr>
<td>CH10</td>
<td>R</td>
</tr>
<tr>
<td>CH11</td>
<td>R</td>
</tr>
<tr>
<td>CH12</td>
<td>R</td>
</tr>
<tr>
<td>CH13</td>
<td>R</td>
</tr>
<tr>
<td>CH14</td>
<td>R</td>
</tr>
<tr>
<td>CH15</td>
<td>R</td>
</tr>
</tbody>
</table>

#### Name

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNTOF</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

#### Bit Value \( \times \) Name \( \times \) Reset \( \times \) Access \( \times \) Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>CNTOF</td>
<td>0</td>
<td>R</td>
<td>Set when the LESENSE counter overflows.</td>
</tr>
</tbody>
</table>

Set when the result buffer overflows.
<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>BUFLEVEL</td>
<td>0</td>
<td>R</td>
<td>Set when the data buffer is full.</td>
</tr>
<tr>
<td>19</td>
<td>BUFDATAV</td>
<td>0</td>
<td>R</td>
<td>Set when data is available in the result buffer.</td>
</tr>
<tr>
<td>18</td>
<td>DECERR</td>
<td>0</td>
<td>R</td>
<td>Set when the decoder detects an error</td>
</tr>
<tr>
<td>17</td>
<td>DEC</td>
<td>0</td>
<td>R</td>
<td>Set when the decoder has issued and interrupt request</td>
</tr>
<tr>
<td>16</td>
<td>SCANCOMPLETE</td>
<td>0</td>
<td>R</td>
<td>Set when a scan sequence is completed</td>
</tr>
<tr>
<td>15</td>
<td>CH15</td>
<td>0</td>
<td>R</td>
<td>Set when channel 15 triggers</td>
</tr>
<tr>
<td>14</td>
<td>CH14</td>
<td>0</td>
<td>R</td>
<td>Set when channel 14 triggers</td>
</tr>
<tr>
<td>13</td>
<td>CH13</td>
<td>0</td>
<td>R</td>
<td>Set when channel 13 triggers</td>
</tr>
<tr>
<td>12</td>
<td>CH12</td>
<td>0</td>
<td>R</td>
<td>Set when channel 12 triggers</td>
</tr>
<tr>
<td>11</td>
<td>CH11</td>
<td>0</td>
<td>R</td>
<td>Set when channel 11 triggers</td>
</tr>
<tr>
<td>10</td>
<td>CH10</td>
<td>0</td>
<td>R</td>
<td>Set when channel 10 triggers</td>
</tr>
<tr>
<td>9</td>
<td>CH9</td>
<td>0</td>
<td>R</td>
<td>Set when channel 9 triggers</td>
</tr>
<tr>
<td>8</td>
<td>CH8</td>
<td>0</td>
<td>R</td>
<td>Set when channel 8 triggers</td>
</tr>
<tr>
<td>7</td>
<td>CH7</td>
<td>0</td>
<td>R</td>
<td>Set when channel 7 triggers</td>
</tr>
<tr>
<td>6</td>
<td>CH6</td>
<td>0</td>
<td>R</td>
<td>Set when channel 6 triggers</td>
</tr>
<tr>
<td>5</td>
<td>CH5</td>
<td>0</td>
<td>R</td>
<td>Set when channel 5 triggers</td>
</tr>
<tr>
<td>4</td>
<td>CH4</td>
<td>0</td>
<td>R</td>
<td>Set when channel 4 triggers</td>
</tr>
<tr>
<td>3</td>
<td>CH3</td>
<td>0</td>
<td>R</td>
<td>Set when channel 3 triggers</td>
</tr>
<tr>
<td>2</td>
<td>CH2</td>
<td>0</td>
<td>R</td>
<td>Set when channel 2 triggers</td>
</tr>
<tr>
<td>1</td>
<td>CH1</td>
<td>0</td>
<td>R</td>
<td>Set when channel 1 triggers</td>
</tr>
<tr>
<td>0</td>
<td>CH0</td>
<td>0</td>
<td>R</td>
<td>Set when channel 0 triggers</td>
</tr>
</tbody>
</table>
## 21.5.18 LESENSE_IFC - Interrupt Flag Clear Register

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x044</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reset</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
<tr>
<td>W1</td>
<td>W1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:32:31</td>
<td>Reserved</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
<td></td>
</tr>
</tbody>
</table>

23 **CNTOF**
- Write to 1 to clear CNTOF interrupt flag

22 **BUFOF**
- Write to 1 to clear BUFOF interrupt flag

21 **BUFLEVEL**
- Write to 1 to clear BUFLEVEL interrupt flag

20 **BUFDATAV**
- Write to 1 to clear BUFDATAV interrupt flag

19 **DECERR**
- Write to 1 to clear DECERR interrupt flag

18 **DEC**
- Write to 1 to clear DEC interrupt flag

17 **SCANCOMPLETE**
- Write to 1 to clear SCANCOMPLETE interrupt flag

16 **CH15**
- Write to 1 to clear CH15 interrupt flag

15 **CH14**
- Write to 1 to clear CH14 interrupt flag

14 **CH13**
- Write to 1 to clear CH13 interrupt flag

13 **CH12**
- Write to 1 to clear CH12 interrupt flag

12 **CH11**
- Write to 1 to clear CH11 interrupt flag

11 **CH10**
- Write to 1 to clear CH10 interrupt flag

10 **CH9**
- Write to 1 to clear CH9 interrupt flag

9 **CH8**
- Write to 1 to clear CH8 interrupt flag

8 **CH7**
- Write to 1 to clear CH7 interrupt flag

7 **CH6**
- Write to 1 to clear CH6 interrupt flag
### Bit Name Reset Access Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>CH5</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH5 interrupt flag</td>
</tr>
<tr>
<td>4</td>
<td>CH4</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH4 interrupt flag</td>
</tr>
<tr>
<td>3</td>
<td>CH3</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH3 interrupt flag</td>
</tr>
<tr>
<td>2</td>
<td>CH2</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH2 interrupt flag</td>
</tr>
<tr>
<td>1</td>
<td>CH1</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH1 interrupt flag</td>
</tr>
<tr>
<td>0</td>
<td>CH0</td>
<td>0</td>
<td>W1</td>
<td>Write to 1 to clear CH0 interrupt flag</td>
</tr>
</tbody>
</table>

### 21.5.19 LESENSE_IFS - Interrupt Flag Set Register

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x048</td>
<td></td>
</tr>
</tbody>
</table>

**Reset**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>CNTOF</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>BUFOF</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>BUFLEVEL</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>BUFDATAV</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>DECERR</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>DEC</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>SCANCOMPLETE</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>CH15</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>CH14</td>
<td>0</td>
</tr>
</tbody>
</table>

**Access**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>CNTOF</td>
<td>W1</td>
</tr>
<tr>
<td>30</td>
<td>BUFOF</td>
<td>W1</td>
</tr>
<tr>
<td>29</td>
<td>BUFLEVEL</td>
<td>W1</td>
</tr>
<tr>
<td>28</td>
<td>BUFDATAV</td>
<td>W1</td>
</tr>
<tr>
<td>27</td>
<td>DECERR</td>
<td>W1</td>
</tr>
<tr>
<td>26</td>
<td>DEC</td>
<td>W1</td>
</tr>
<tr>
<td>25</td>
<td>SCANCOMPLETE</td>
<td>W1</td>
</tr>
<tr>
<td>24</td>
<td>CH15</td>
<td>W1</td>
</tr>
<tr>
<td>23</td>
<td>CH14</td>
<td>W1</td>
</tr>
</tbody>
</table>

**Description**

31:23: **Reserved**

To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

22:14: **Write to 1 to set the CNTOF interrupt flag**

21:14: **Write to 1 to set the BUFLEVEL interrupt flag**

20:14: **Write to 1 to set the BUFDATAV interrupt flag**

19:14: **Write to 1 to set the DEC interrupt flag**

18:14: **Write to 1 to set the DECERR interrupt flag**

17:14: **Write to 1 to set the SCANCOMPLETE interrupt flag**

16:14: **Write to 1 to set the CH15 interrupt flag**

15:14: **Write to 1 to set the CH14 interrupt flag**

14:14: **Write to 1 to set the CH13 interrupt flag**

13:14: **Write to 1 to set the CH12 interrupt flag**

12:14: **Write to 1 to set the CH11 interrupt flag**

11:14: **Write to 1 to set the CH10 interrupt flag**

10:14: **Write to 1 to set the CH9 interrupt flag**

9:14: **Write to 1 to set the CH8 interrupt flag**

8:14: **Write to 1 to set the CH7 interrupt flag**

7:14: **Write to 1 to set the CH6 interrupt flag**

6:14: **Write to 1 to set the CH5 interrupt flag**

5:14: **Write to 1 to set the CH4 interrupt flag**

4:14: **Write to 1 to set the CH3 interrupt flag**

3:14: **Write to 1 to set the CH2 interrupt flag**

2:14: **Write to 1 to set the CH1 interrupt flag**

1:14: **Write to 1 to set the CH0 interrupt flag**

0:14: **Write to 1 to set the CH0 interrupt flag**
## 21.5.20 LESENSE_IEN - Interrupt Enable Register

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x04C</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:23</td>
<td>Reserved</td>
<td>0</td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>22</td>
<td>CNTOF</td>
<td>0</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td>Bit</td>
<td>Name</td>
<td>Reset</td>
<td>Access</td>
<td>Description</td>
</tr>
<tr>
<td>-----</td>
<td>----------</td>
<td>-------</td>
<td>--------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>21</td>
<td>BUFOF</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the BUFOF interrupt flag</td>
</tr>
<tr>
<td>20</td>
<td>BUFLEVEL</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the BUFLEVEL interrupt flag</td>
</tr>
<tr>
<td>19</td>
<td>BUFDATAV</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the BUFDATAV interrupt flag</td>
</tr>
<tr>
<td>18</td>
<td>DECERR</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the DECERR interrupt flag</td>
</tr>
<tr>
<td>17</td>
<td>DEC</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the DEC interrupt flag</td>
</tr>
<tr>
<td>16</td>
<td>SCANCOMPLETE</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the SCANCOMPLETE interrupt flag</td>
</tr>
<tr>
<td>15</td>
<td>CH15</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH15 interrupt flag</td>
</tr>
<tr>
<td>14</td>
<td>CH14</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH14 interrupt flag</td>
</tr>
<tr>
<td>13</td>
<td>CH13</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH13 interrupt flag</td>
</tr>
<tr>
<td>12</td>
<td>CH12</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH12 interrupt flag</td>
</tr>
<tr>
<td>11</td>
<td>CH11</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH11 interrupt flag</td>
</tr>
<tr>
<td>10</td>
<td>CH10</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH10 interrupt flag</td>
</tr>
<tr>
<td>9</td>
<td>CH9</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH9 interrupt flag</td>
</tr>
<tr>
<td>8</td>
<td>CH8</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH8 interrupt flag</td>
</tr>
<tr>
<td>7</td>
<td>CH7</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH7 interrupt flag</td>
</tr>
<tr>
<td>6</td>
<td>CH6</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH6 interrupt flag</td>
</tr>
<tr>
<td>5</td>
<td>CH5</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH5 interrupt flag</td>
</tr>
<tr>
<td>4</td>
<td>CH4</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH4 interrupt flag</td>
</tr>
<tr>
<td>3</td>
<td>CH3</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH3 interrupt flag</td>
</tr>
<tr>
<td>2</td>
<td>CH2</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH2 interrupt flag</td>
</tr>
<tr>
<td>1</td>
<td>CH1</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH1 interrupt flag</td>
</tr>
<tr>
<td>0</td>
<td>CH0</td>
<td>0</td>
<td>RW</td>
<td>Set to enable interrupt on the CH0 interrupt flag</td>
</tr>
</tbody>
</table>
## 21.5.21 LESENSE_SYNCBUSY - Synchronization Busy Register

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x050</td>
<td></td>
</tr>
</tbody>
</table>

### Reset
- 0

### Access
- R

### Name
- EVAL
- INTERACT
- TIMING
- DATA
- TCONFB
- TCONFA
- RIPCNT
- TESTCTRL
- FEATURECONF
- POWERDOWN
- ROUTE
- ALTEXCONF
- SENSORSTATE
- DECSATE
- CURCH
- BUFDATA
- FPR
- STATUS
- SCANRES
- CHEN
- CMD
- BIASCTRL
- DECCTRL
- PERCTRL
- TIMCTRL
- CTRL

### Bit | Name | Reset | Access | Description
---|------|-------|--------|--------------------------------------------------
31:27 | Reserved | 0 | R | To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)
26 | EVAL | 0 | R | LESENSE_CHx_EVAL Register Busy
\[\text{Set when the value written to LESENSE_CHx_EVAL is being synchronized.}\]
25 | INTERACT | 0 | R | LESENSE_CHx_INTERACT Register Busy
\[\text{Set when the value written to LESENSE_CHx_INTERACT is being synchronized.}\]
24 | TIMING | 0 | R | LESENSE_CHx_TIMING Register Busy
\[\text{Set when the value written to LESENSE_CHx_TIMING is being synchronized.}\]
23 | DATA | 0 | R | LESENSE_BUFx_DATA Register Busy
\[\text{Set when the value written to LESENSE_BUFx_DATA is being synchronized.}\]
22 | TCONFB | 0 | R | LESENSE_STx_TCONFB Register Busy
\[\text{Set when the value written to LESENSE_STx_TCONFB is being synchronized.}\]
21 | TCONFA | 0 | R | LESENSE_STx_TCONFA Register Busy
\[\text{Set when the value written to LESENSE_STx_TCONFA is being synchronized.}\]
20 | RIPCNT | 0 | R | LESENSE_RIPCNT Register Busy
\[\text{Set when the value written to LESENSE_RIPCNT is being synchronized.}\]
19 | TESTCTRL | 0 | R | LESENSE_TESTCTRL Register Busy
\[\text{Set when the value written to LESENSE_TESTCTRL is being synchronized.}\]
18 | FEATURECONF | 0 | R | LESENSE_FEATURECONF Register Busy
\[\text{Set when the value written to LESENSE_FEATURECONF is being synchronized.}\]
17 | POWERDOWN | 0 | R | LESENSE_POWERDOWN Register Busy
\[\text{Set when the value written to LESENSE_POWERDOWN is being synchronized.}\]
16 | ROUTE | 0 | R | LESENSE_ROUTE Register Busy
\[\text{Set when the value written to LESENSE_ROUTE is being synchronized.}\]
15 | ALTEXCONF | 0 | R | LESENSE_ALTEXCONF Register Busy
\[\text{Set when the value written to LESENSE_ALTEXCONF is being synchronized.}\]
14 | IDLECONF | 0 | R | LESENSE_IDLECONF Register Busy
\[\text{Set when the value written to LESENSE_IDLECONF is being synchronized.}\]
13 | SENSORSTATE | 0 | R | LESENSE_SENSORSTATE Register Busy
\[\text{Set when the value written to LESENSE_SENSORSTATE is being synchronized.}\]
12 | DECSATE | 0 | R | LESENSE_DECSTATE Register Busy
\[\text{Set when the value written to LESENSE_DECSTATE is being synchronized.}\]
11 | CURCH | 0 | R | LESENSE_CURCH Register Busy
\[\text{Set when the value written to LESENSE_CURCH is being synchronized.}\]
10 | BUFDATA | 0 | R | LESENSE_BUFDATA Register Busy
\[\text{Set when the value written to LESENSE_BUFDATA is being synchronized.}\]
<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Reserved</td>
<td></td>
<td>RW</td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>23</td>
<td>ALTEX7PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX7 Pin Enable</td>
</tr>
<tr>
<td>22</td>
<td>ALTEX6PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX6 Pin Enable</td>
</tr>
<tr>
<td>21</td>
<td>ALTEX5PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX5 Pin Enable</td>
</tr>
<tr>
<td>20</td>
<td>ALTEX4PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX4 Pin Enable</td>
</tr>
<tr>
<td>19</td>
<td>ALTEX3PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX3 Pin Enable</td>
</tr>
</tbody>
</table>

### 21.5.22 LESENSE_ROUTE - I/O Routing Register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x054</td>
<td>31:24</td>
<td></td>
<td></td>
<td>Reserved</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>ALTEX7PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX7 Pin Enable</td>
</tr>
<tr>
<td>22</td>
<td>ALTEX6PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX6 Pin Enable</td>
</tr>
<tr>
<td>21</td>
<td>ALTEX5PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX5 Pin Enable</td>
</tr>
<tr>
<td>20</td>
<td>ALTEX4PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX4 Pin Enable</td>
</tr>
<tr>
<td>19</td>
<td>ALTEX3PEN</td>
<td>0</td>
<td>RW</td>
<td>ALTEX3 Pin Enable</td>
</tr>
</tbody>
</table>

*This document is a sample content and does not reflect the actual content of the original document.*
### Bit | Name | Reset | Access | Description
--- | --- | --- | --- | ---
18 | ALTEX2PEN | 0 | RW | ALTEX2 Pin Enable
17 | ALTEX1PEN | 0 | RW | ALTEX1 Pin Enable
16 | ALTEX0PEN | 0 | RW | ALTEX0 Pin Enable
15 | CH15PEN | 0 | RW | CH15 Pin Enable
14 | CH14PEN | 0 | RW | CH14 Pin Enable
13 | CH13PEN | 0 | RW | CH13 Pin Enable
12 | CH12PEN | 0 | RW | CH12 Pin Enable
11 | CH11PEN | 0 | RW | CH11 Pin Enable
10 | CH10PEN | 0 | RW | CH10 Pin Enable
9 | CH9PEN | 0 | RW | CH9 Pin Enable
8 | CH8PEN | 0 | RW | CH8 Pin Enable
7 | CH7PEN | 0 | RW | CH7 Pin Enable
6 | CH6PEN | 0 | RW | CH6 Pin Enable
5 | CH5PEN | 0 | RW | CH5 Pin Enable
4 | CH4PEN | 0 | RW | CH4 Pin Enable
3 | CH3PEN | 0 | RW | CH3 Pin Enable
2 | CH2PEN | 0 | RW | CH2 Pin Enable
1 | CH1PEN | 0 | RW | CH0 Pin Enable
0 | CH0PEN | 0 |RW | CH0 Pin Enable

### 21.5.23 LESENSE_POWERDOWN - LESENSE RAM power-down register (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).
### 21.5.24 LESENSE_STx_TCONFA - State transition configuration A (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x200</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

### Description

- **31:19 Reserved**: To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

- **18 CHAIN**: Enable state descriptor chaining. When set, descriptor in the next location will also be evaluated.

- **17 Reserved**: To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

- **16 SETIF**: Set interrupt flag enable. Set interrupt flag when sensor state equals COMP.

- **15 Reserved**: To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

- **14:12 PRSACT**: Configure transition action. Configure which action to perform when sensor state equals COMP.

#### PRSCNT = 0

<table>
<thead>
<tr>
<th>Mode</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>0</td>
<td>No PRS pulses generated</td>
</tr>
<tr>
<td>PRS0</td>
<td>1</td>
<td>Generate pulse on LESPRS0</td>
</tr>
<tr>
<td>PRS1</td>
<td>2</td>
<td>Generate pulse on LESPRS1</td>
</tr>
<tr>
<td>PRS01</td>
<td>3</td>
<td>Generate pulse on LESPRS0 and LESPRS1</td>
</tr>
<tr>
<td>PRS2</td>
<td>4</td>
<td>Generate pulse on LESPRS2</td>
</tr>
<tr>
<td>PRS02</td>
<td>5</td>
<td>Generate pulse on LESPRS0 and LESPRS2</td>
</tr>
<tr>
<td>PRS12</td>
<td>6</td>
<td>Generate pulse on LESPRS1 and LESPRS2</td>
</tr>
<tr>
<td>PRS012</td>
<td>7</td>
<td>Generate pulse on LESPRS0, LESPRS1 and LESPRS2</td>
</tr>
</tbody>
</table>

#### PRSCNT = 1
---the world's most energy friendly microcontrollers

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRSCNT = 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NONE</td>
<td></td>
<td>0</td>
<td>Do not count</td>
<td></td>
</tr>
<tr>
<td>UP</td>
<td></td>
<td>1</td>
<td>Count up</td>
<td></td>
</tr>
<tr>
<td>DOWN</td>
<td></td>
<td>2</td>
<td>Count down</td>
<td></td>
</tr>
<tr>
<td>PRS2</td>
<td></td>
<td>4</td>
<td>Generate pulse on LESPRS2</td>
<td></td>
</tr>
<tr>
<td>UPANDPRS2</td>
<td></td>
<td>5</td>
<td>Count up and generate pulse on LESPRS2</td>
<td></td>
</tr>
<tr>
<td>DOWNANDPRS2</td>
<td></td>
<td>6</td>
<td>Count down and generate pulse on LESPRS2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>11:8</th>
<th>NEXTSTATE</th>
<th>0xX</th>
<th>RW</th>
<th>Next state index</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Index of next state to be entered if the sensor state equals COMP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7:4</th>
<th>MASK</th>
<th>0xX</th>
<th>RW</th>
<th>Sensor mask</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Set bit X to exclude sensor X from evaluation.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3:0</th>
<th>COMP</th>
<th>0xX</th>
<th>RW</th>
<th>Sensor compare value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>State transition is triggered when sensor state equals COMP</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 21.5.25 LESENSE_STx_TCONFb - State transition configuration B (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x204</td>
<td>31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>Reset</td>
<td>X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X 00X</td>
</tr>
<tr>
<td>Access</td>
<td>RW RW RW RW RW RW RW</td>
</tr>
<tr>
<td>Name</td>
<td>SETIF PRSACT NEXTSTATE MASK COMP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:17</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>16</td>
<td>SETIF</td>
<td>X</td>
<td>RW</td>
<td>Set interrupt flag</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Set interrupt flag when sensor state equals COMP</td>
</tr>
<tr>
<td>15</td>
<td>Reserved</td>
<td></td>
<td></td>
<td>To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)</td>
</tr>
<tr>
<td>14:12</td>
<td>PRSACT</td>
<td>0xX</td>
<td>RW</td>
<td>Configure transition action</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Configure which action to perform when sensor state equals COMP</td>
</tr>
</tbody>
</table>

<p>| PRSCNT = 0 |</p>
<table>
<thead>
<tr>
<th>Mode</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>0</td>
<td>No PRS pulses generated</td>
</tr>
<tr>
<td>PRS0</td>
<td>1</td>
<td>Generate pulse on PRS0</td>
</tr>
<tr>
<td>PRS1</td>
<td>2</td>
<td>Generate pulse on PRS1</td>
</tr>
<tr>
<td>PRS01</td>
<td>3</td>
<td>Generate pulse on PRS0 and PRS1</td>
</tr>
<tr>
<td>PRS2</td>
<td>4</td>
<td>Generate pulse on PRS2</td>
</tr>
<tr>
<td>PRS02</td>
<td>5</td>
<td>Generate pulse on PRS0 and PRS2</td>
</tr>
<tr>
<td>PRS12</td>
<td>6</td>
<td>Generate pulse on PRS1 and PRS2</td>
</tr>
<tr>
<td>PRS012</td>
<td>7</td>
<td>Generate pulse on PRS0, PRS1 and PRS2</td>
</tr>
</tbody>
</table>

<p>| PRSCNT = 1 |</p>
<table>
<thead>
<tr>
<th>Mode</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NONE</td>
<td>0</td>
<td>Do not count</td>
</tr>
<tr>
<td>UP</td>
<td>1</td>
<td>Count up</td>
</tr>
<tr>
<td>DOWN</td>
<td>2</td>
<td>Count down</td>
</tr>
<tr>
<td>PRS2</td>
<td>4</td>
<td>Generate pulse on PRS2</td>
</tr>
</tbody>
</table>
### 21.5.26 LESENSE_BUFx_DATA - Scan results (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset (0x280)</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td></td>
</tr>
<tr>
<td>24:16</td>
<td></td>
</tr>
<tr>
<td>15:0</td>
<td>DATA</td>
</tr>
</tbody>
</table>

#### Bit Position

- **Reset**: To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>DATA</td>
<td>0xFFFF</td>
<td>RW</td>
<td>Scan result buffer</td>
</tr>
</tbody>
</table>

### 21.5.27 LESENSE_CHx_TIMING - Scan configuration (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset (0x2C0)</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td></td>
</tr>
<tr>
<td>24:16</td>
<td></td>
</tr>
<tr>
<td>19:0</td>
<td></td>
</tr>
</tbody>
</table>

#### Bit Position

- **Reset**: To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3)

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:20</td>
<td>Reserved</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19:13</td>
<td>MEASUREDLY</td>
<td>0xXX</td>
<td>RW</td>
<td>Set measure delay</td>
</tr>
</tbody>
</table>

Configure measure delay. Sensor measuring is delayed for MEASUREDLY+1 EXCLK cycles.
### Bit Name Reset Access Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Reset</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:6</td>
<td>SAMPLEDLY</td>
<td>0xXX</td>
<td>RW</td>
<td>Set sample delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Configure sample delay. Sampling will occur after SAMPLEDLY+1 SAMPLECLK cycles.</td>
</tr>
<tr>
<td>5:0</td>
<td>EXTIME</td>
<td>0xXX</td>
<td>RW</td>
<td>Set excitation time</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Configure excitation time. Excitation will last EXTIME+1 EXCLK cycles.</td>
</tr>
</tbody>
</table>

#### 21.5.28 LESENSE_CHx_INTERACT - Scan configuration (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

### Offset Bit Position

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2C4</td>
<td></td>
</tr>
</tbody>
</table>

#### Reset

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>31:0</td>
<td>Reserved</td>
<td>RW</td>
<td></td>
</tr>
</tbody>
</table>

To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

#### Access

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>ALTEX</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Use alternative excite pin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>If set, alternative excite pin will be used for excitation</td>
</tr>
<tr>
<td>18</td>
<td>SAMPLECLK</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Select clock used for timing of sample delay</td>
</tr>
</tbody>
</table>

#### Name

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>EXCLkHz</td>
<td>RW</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Select clock used for excitation timing</td>
</tr>
</tbody>
</table>

#### Value Mode Description

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>LFACLK</td>
<td>LFACLK will be used for timing</td>
</tr>
<tr>
<td>1</td>
<td>AUXHFRCO</td>
<td>AUXHFRCO will be used for timing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
<th>Access</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>16:15</td>
<td>EXMODE</td>
<td>0xX</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Set GPIO mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>GPIO mode for the excitation phase of the scan sequence. Note that DACOUT is only available on channels 0, 1, 2, 3, 12, 13, 14, and 15.</td>
</tr>
</tbody>
</table>

#### Value Mode Description

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DISABLE</td>
<td>Disabled</td>
</tr>
<tr>
<td>1</td>
<td>HIGH</td>
<td>Push Pull, GPIO is driven high</td>
</tr>
<tr>
<td>2</td>
<td>LOW</td>
<td>Push Pull, GPIO is driven low</td>
</tr>
<tr>
<td>3</td>
<td>DACOUT</td>
<td>DAC output</td>
</tr>
</tbody>
</table>

#### 14:13 SETIF 0xX RW Enable interrupt generation

Select interrupt generation mode for CHx interrupt flag.

#### Value Mode Description

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>NONE</td>
<td>No interrupt is generated</td>
</tr>
<tr>
<td>1</td>
<td>LEVEL</td>
<td>Set interrupt flag if the sensor triggers.</td>
</tr>
<tr>
<td>2</td>
<td>POSEDGE</td>
<td>Set interrupt flag on positive edge on the sensor state</td>
</tr>
<tr>
<td>3</td>
<td>NEGEDGE</td>
<td>Set interrupt flag on negative edge on the sensor state</td>
</tr>
</tbody>
</table>
### 21.5.29 LESENSE_CHx_EVAL - Scan configuration (Async Reg)

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

#### Offset Bit Position

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2C8</td>
<td>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X X X X X X 0xXXXX</td>
</tr>
</tbody>
</table>

**Access**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW RW RW RW RW RW RW</td>
</tr>
</tbody>
</table>

**Name**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCANRESINV STRSAMPLE DECODE COMP COMPTHRES</td>
</tr>
</tbody>
</table>

#### Bit 12

**Name**: SAMPLE

- **Reset**: X
- **Access**: RW
- **Description**: Select sample mode

Select if ACMP output or counter output should be used in comparison.

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>COUNTER</td>
<td>Counter output will be used in comparison</td>
</tr>
<tr>
<td>1</td>
<td>ACMP</td>
<td>ACMP output will be used in comparison</td>
</tr>
</tbody>
</table>

#### Bit 11:0

**Name**: ACMPTHRES

- **Reset**: 0xXXX
- **Access**: RW
- **Description**: Set ACMP threshold

Select ACMP threshold.

---

**21.5.29 LESENSE_CHx_EVAL - Scan configuration (Async Reg)**

For more information about Asynchronous Registers please see Section 5.3 (p. 18).

<table>
<thead>
<tr>
<th>Offset</th>
<th>Bit Position</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x2C8</td>
<td>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

**Reset**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>X X X X X X 0xXXXX</td>
</tr>
</tbody>
</table>

**Access**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RW RW RW RW RW RW RW</td>
</tr>
</tbody>
</table>

**Name**

<table>
<thead>
<tr>
<th>31 30 29 28 27 26 25 24 23 22 21 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCANRESINV STRSAMPLE DECODE COMP COMPTHRES</td>
</tr>
</tbody>
</table>

**Bit 31:20 Reserved**

To ensure compatibility with future devices, always write bits to 0. More information in Section 2.1 (p. 3).

**Bit 19**

**Name**: SCANRESINV

- **Reset**: X
- **Access**: RW
- **Description**: Enable inversion of result

If set, the bit stored in SCANRES will be inverted.

**Bit 18**

**Name**: STRSAMPLE

- **Reset**: X
- **Access**: RW
- **Description**: Select if counter result should be stored

If set, the counter value will be stored and available in the result buffer.

**Bit 17**

**Name**: DECODE

- **Reset**: X
- **Access**: RW
- **Description**: Send result to decoder

If set, the result from this channel will be shifted into the decoder register.

**Bit 16**

**Name**: COMP

- **Reset**: X
- **Access**: RW
- **Description**: Select mode for counter comparison

Set compare mode

<table>
<thead>
<tr>
<th>Value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>LESS</td>
<td>Set interrupt flag if counter value is less than CTRTHRESHOLD, or if the ACMP output is 0</td>
</tr>
<tr>
<td>1</td>
<td>GE</td>
<td>Set interrupt flag if counter value is greater than, or equal to CTRTHRESHOLD, or if the ACMP output is 1</td>
</tr>
</tbody>
</table>

**Bit 15:0**

**Name**: COMPTHRES

- **Reset**: 0xXXXX
- **Access**: RW
- **Description**: Decision threshold for counter

Set counter threshold
22 ACMP - Analog Comparator

Quick Facts

What?
The ACMP (Analog Comparator) compares two analog signals and returns a digital value telling which is greater.

Why?
Applications often do not need to know the exact value of an analog signal, only if it has passed a certain threshold. Often the voltage must be monitored continuously, which requires extremely low power consumption.

How?
Available down to Energy Mode 3 and using as little as 100 nA, the ACMP can wake up the system when input signals pass the threshold. The analog comparator can compare two analog signals or one analog signal and a highly configurable internal reference.

22.1 Introduction

The Analog Comparator is used to compare the voltage of two analog inputs, with a digital output indicating which input voltage is higher. Inputs can either be one of the selectable internal references or from external pins. Response time and thereby also the current consumption can be configured by altering the current supply to the comparator.

22.2 Features

- 8 selectable external positive inputs
- 8 selectable external negative inputs
- 5 selectable internal negative inputs
  - Internal 1.25 V bandgap
  - Internal 2.5 V bandgap
  - \( V_{DD} \) scaled by 64 selectable factors
  - DAC channel 0 and 1
- Low power mode for internal \( V_{DD} \) and bandgap references
- Selectable hysteresis
  - 8 levels between 0 and ±70 mV
- Selectable response time
- Asynchronous interrupt generation on selectable edges
  - Rising edge
  - Falling edge
  - Both edges
- Operational in EM0-EM3
- Dedicated capacitive sense mode with up to 8 inputs
  - Adjustable internal resistor
  - Configurable inversion of comparator output