HAL version deprecation

In the L release of Android, we are halting support for some sensor HAL versions. The only supported versions are `SENSORS_DEVICE_API_VERSION_1_0` and `SENSORS_DEVICE_API_VERSION_1_3`.

In the next releases, we are likely to drop support for 1_0 as well.

1_0 has no concept of batching. If possible, all devices using 1_0 SHOULD upgrade to 1_3.

1_1 and 1_2 suffer from poor definition of the batching concept, and are not supported anymore

All devices currently using 1_1 or 1_2 MUST upgrade to 1_3.

In 1_3, we simplified the notion of batching, and we introduced wake up sensors.

To upgrade to 1_3, follow the changes listed below.

Implement the batch function

Even if you do not implement batching (your hardware has no FIFO), you must implement the **`batch`** function. **`batch`** is used to set the sampling period and the maximum reporting latency for a given sensor. It replaces **`setDelay`**. **`setDelay`** will not be called anymore.

If you do not implement batching, you can implement **`batch`** by simply calling your existing **`setDelay`** function with the provided **`sampling_period_ns`** parameter.

Implement the flush function

Even if you do not implement batching, you must implement the **`flush`** function.

If you do not implement batching, **`flush`** must generate one **`META_DATA_FLUSH_COMPLETE`** event and return 0 (success).

Change your sensors_poll_device_t.common.version

```
your_poll_device.common.version = SENSORS_DEVICE_API_VERSION_1_3
```

Add the new fields to the definition of your sensors

When defining each sensor, in addition to the usual sensor_t (/devices/sensors/hal-interface.html#sensor_t) fields:

```
.name =       "My magnetic field Sensor",
.vendor =     "My company",
.version
=    1,
.handle =     mag_handle,
.type =       SENSOR_TYPE_MAGNETIC_FIELD,
.maxRange =   200.0f,
.resolution = CONVERT_M,
.power =      5.0f,
.minDelay =
 16667,
```

you also must set the new fields, defined between 1_0 and 1_3:

```
.fifoReservedEventCount = 0,
.fifoMaxEventCount =   0,
.stringType =         0,
.requiredPermission = 0,
.maxDelay =      200000
.flags =
SENSOR_FLAG_CONTINUOUS_MODE,
```

fifoReservedEventCount: If not implementing batching, set this one to 0.

fifoMaxEventCount: If not implementing batching, set this one to 0

stringType: Set to 0 for all official android sensors (those that are defined in sensors.h), as this value will be overwritten by the framework. For non-official sensors, see sensor_t (/devices/sensors/hal-interface.html#sensor_t) for details on how to set it.

requiredPermission: This is the permission that applications will be required to have to get access to your sensor. You can usually set this to 0 for all of your sensors, but sensors with type `HEART_RATE` must set this to `SENSOR_PERMISSION_BODY_SENSORS.`

maxDelay: This value is important and you will need to set it according to the capabilities of the sensor and of its driver.

This value is defined only for continuous and on-change sensors. It is the delay between two sensor events corresponding to the lowest frequency that this sensor supports. When lower frequencies are requested through the `batch` function, the events will be generated at this frequency instead. It can be used by the framework or applications to estimate when the batch FIFO may be full. If this value is not set properly, CTS will fail. For one-shot and special reporting mode sensors, set `maxDelay` to 0.

For continuous sensors, set it to the maximum sampling period allowed in microseconds.

The following are applicable for `period_ns`, `maxDelay`, and `minDelay`:

- `period_ns` is in nanoseconds whereas `maxDelay`/`minDelay` are in microseconds.
- `maxDelay` should always fit within a 32-bit signed integer. It is declared as 64-bit on 64-bit architectures only for binary compatibility reasons.

flags: This field defines the reporting mode of the sensor and whether the sensor is a wake up sensor.

If you do not implement batching, and are just moving from 1.0 to 1.3, set this to:

`SENSOR_FLAG_WAKE_UP | SENSOR_FLAG_ONE_SHOT_MODE` for one-shot (/devices/sensors/report-modes.html#one-shot) sensors

`SENSOR_FLAG_CONTINUOUS_MODE` for continuous (/devices/sensors/report-modes.html#continuous) sensors `SENSOR_FLAG_ON_CHANGE_MODE` for on-change (/devices/sensors/report-modes.html#on-change) sensors except proximity (#proximity) `SENSOR_FLAG_SPECIAL_REPORTING_MODE` for sensors with special (/devices/sensors/report-modes.html#special) reporting mode except for the tilt detector (/devices/sensors/sensor-types.html#tilt_detector).

`SENSOR_FLAG_WAKE_UP | SENSOR_FLAG_ON_CHANGE_MODE` for the proximity (/devices/sensors/sensor-types.html#proximity) sensor and the Android official tilt detector (/devices/sensors/sensor-types.html#tilt_detector) sensor.

Notes when upgrading from 1_1 or 1_2

- The `batch` function now nearly-always succeeds, even for sensors that do not support batching, independent of the value of the timeout argument. The only cases where the `batch` function might fail are internal errors, or a bad `sensor_handle`, or negative `sampling_period_ns` or negative `max_report_latency_ns`.

- Whether a sensor supports batching is defined by whether it has a `fifoMaxEventCount` greater than 0. (In previous versions, it was based on the return value of `batch()`.)
- Sensors that support batching are always in what we called the “batch mode” in previous versions: even if the `max_report_latency_ns` parameter is 0, the sensor must still be batched, meaning the events must be stored in the FIFO when the SoC goes to suspend mode.
- The `flags` parameter of the `batch` function is not used anymore. `DRY_RUN` and `WAKE_UPON_FIFO_FULL` are both deprecated, and will never be passed to the `batch` function.
- The batch timeout argument is now referred to as the `max_report_latency` argument.