SYNERTRONIC DESIGNS (PTY) LTD | SYNERTRONIC.CO.ZA

CS-002

API for an InGaAs OEM camera

Most OEM solutions also require a custom software API. This is an example of a custom API for an OEM InGaAs camera.

1. Challenge

We developed a low-speed InGaAs OEM camera for a client. The next step was providing an easy-to-use API to integrate the OEM solution into their software.

2. Solution

All our cameras use a proprietary communication protocol. To simplify communication via USB we provide customized APIs that hide the communication details. The API simplifies device communication and provides an abstraction layer to easily take measurements. The API is written in C++ with a C-style interface. This allows integration into most software environments, including .NET, LabView, Python, Matlab. Above is an example of a typical API.

The client's software only needs to:

- Connect with the camera
- Apply settings (e.g. integration time, averaging)
- Start a measurement
- Retrive measurement results
- Post-process the results and present data in their domain-specific software

```
void INGAASLS_getVersion(int* majorVersion, int* minorVersion);

int INGAASLS_getAvailableDevices(int* deviceCount);

int INGAASLS_initialiseUSBSession();int INGAASLS_closeSession();

int INGAASLS_getDeviceInfo(InsopticsLS_DeviceInfo_t* pDeviceInfo);

int INGAASLS_setIntegrationTime(double ms);

int INGAASLS_setAverageCount(int count);

int INGAASLS_setMeasurementMode(InsopticsLS_MeasurementMode mode);
int INGAASLS_startMeasurement();

int INGAASLS_stopMeasurement();

bool INGAASLS_hasMeasurement(int measurementIndex);

int INGAASLS_waitForMeasurement(int measurementIndex, int timeout_ms);

bool INGAASLS_isBusy();int INGAASLS_waitForDone(int timeout_ms);

void INGAASLSS_Measurement_t_init(InsopticsLS_Measurement_t* pMeasurement);

void INGAASLSS_Measurement_t_free(InsopticsLS_Measurement_t* pMeasurement);

int INGAASLS_getMeasurement(int measurementIndex, InsopticsLS_Measurement_t* pMeasurement);

int INGAASLS_calibrateOffset();

int INGAASLSS_getOffset(double* offset_mv);

void INGAASLS_getLastErrorMessage(char* message);
```

3. Impact

Reduced time-to-market: The client's software team does not need to understand the device protocols and can work on a higher abstraction layer.

Increased relibaility: The API uses our established core backend, which has been debugged over years and provides a robust solution.

Support: We offer continuous support, which includes answering software implementation queries and feature extensions.

Documentation: All our APIs are delivered with high-quality documentation.