Getting Started with Scamp5d

version 1.0

This document provides the basic introduction to the SCAMP vision system. It provides the overview of the system hardware, programming model, and the software programming framework. It introduces various software components and applications needed to develop the SCAMP programs, and provides a basic code example.

1. Introduction

Scamp5d is an embedded vision system, or a *smart camera*, build around the SCAMP-5 *vision chip*, which is a custom integrated circuit, combining the functionality of an *image sensor*, and a *processor array* (Figure 1). This unique device provides low-power and high-performance hardware acceleration of low-level vision algorithms. The fundamental principle of operation is that the compute-intensive sections of a vision algorithm are executed on a *massively-parallel pixel-per-processor SIMD array*. This processor array is integrated into the pixels of the image sensor device, hence the processing occurs *"on-sensor" or "on the focal plane".*

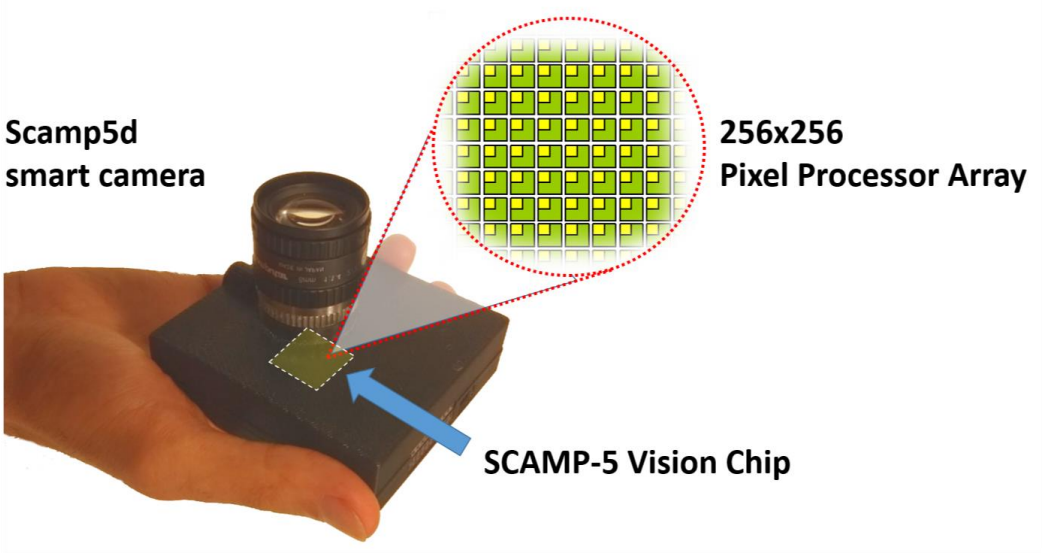

Figure 1. Scamp5d vision system uses a custom image sensor device. The SCAMP-5 chip vision chip combines photosensors (pixels) and processor circuits, providing a massively-parallel on-sensor image processing capability.

The system can operate at very high frame rates (e.g. 1000 fps, and above), processing the image frames internally in the sensor chip. Only the results of these computations are read-out from the chip. In one mode of operation, the outputs can be in the form of *address-events*, representing sparse locations of pixels of interest. Other read-out modes are possible, including a bounding-box coordinates of a region of interest, a binary image, a global measure (a direction of motion, number of object in the image, etc), a sub-window or sub-sampled image, or even several gray-level images (this mode is typically used for debugging purposes).

The Scamp5d smart camera vision system integrates the SCAMP-5 vision sensor chip and additional processor and interface circuits, providing a stand-alone solution to executing vision algorithms. The system can execute autonomously, or it can be interfaced to a host CPU system, for example an embedded computer (e.g. Raspbery Pi, Odroid), or a Windows/Linux PC (this is mostly used for algorithm development/debugging).

2. Hardware Overview

The block diagram of the Scamp5d processing system is shown in Figure 2. The main components of the system are:

SCAMP-5 – vision sensor – this device acts both as a gray-level image sensor, and as a hardware-accelerator of low-level image processing algorithms. It is a custom integrated circuit, designed at the University of Manchester. It comprises a 256x256 pixel array embedding a 65,536-core SIMD processor array. The processor array operates as a "co-processor" to the system processor (MCU).

MCU – microcontroller unit – this device provides the control of the vision sensor, executes the main vision algorithm and interfaces to the external world. The MCU on Scamp5d system is an NXP LPC4357 device, which integrates two 32-bit processor cores: ARM Cortex M0 and ARM Cortex M4, and internal RAM. The function of the processor cores is as follows:

> **M0** – this is the main (although not the most powerful!) processor of the Scamp5d vision system. The vision programs are written in C/C++, compiled for, and executed on this processor core. The hardware-acceleration features of the SCAMP-5 chip are visible to this core as function calls and custom parallel processing *kernels*, through the use of SCAMP Application Programming Interface (API).
>
> **M4** – in the basic configuration this core is only used to provide the interface between the M0/SCAMP vision processor subsystem and the external world. The default firmware is provided, that makes these communications "transparent" in a sense that the user can fully control the operation of the system by using the API provided for programming the M0 core only. Optionally, the user can also program M4 with their own application code, e.g. to implement higher-level vision algorithms, and other system tasks. See Scamp5d as Embedded Vision System for details.

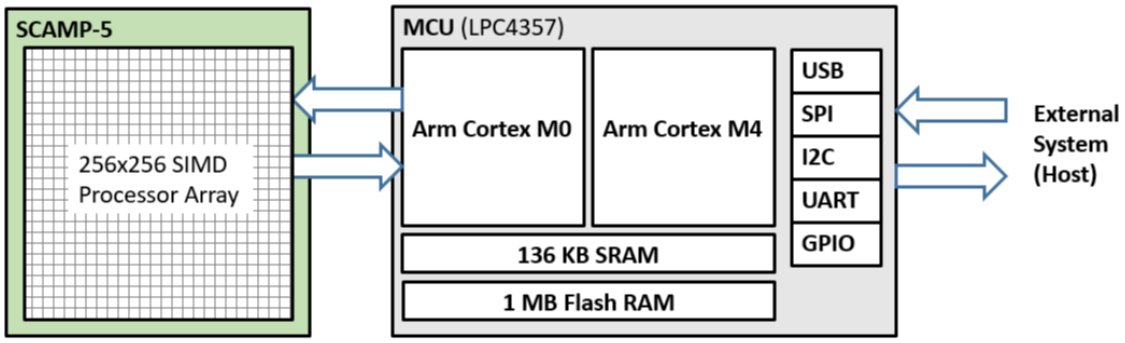

Figure 2. Scamp5d hardware system overview. The M0 core controls the execution of the vision algorithms. The computational power is provided by the pixel-parallel SIMD processor array integrated into the SCAMP-5 vision chip.

The Scamp5d system also includes calibration, clock, and interface circuits. The supported interfaces include USB 2.0, SPI, I^2C, UART, and several general-purpose input/output (GPIO) ports. The MCU includes built-in SRAM (136kB) and Flash (1MB) memory. The Scamp5d Hardware Datasheet provides details of the system hardware, electrical specifications, and interfacing options.

3. Software Overview

The basic structure of the SCAMP software programming framework is shown in Figure 3.

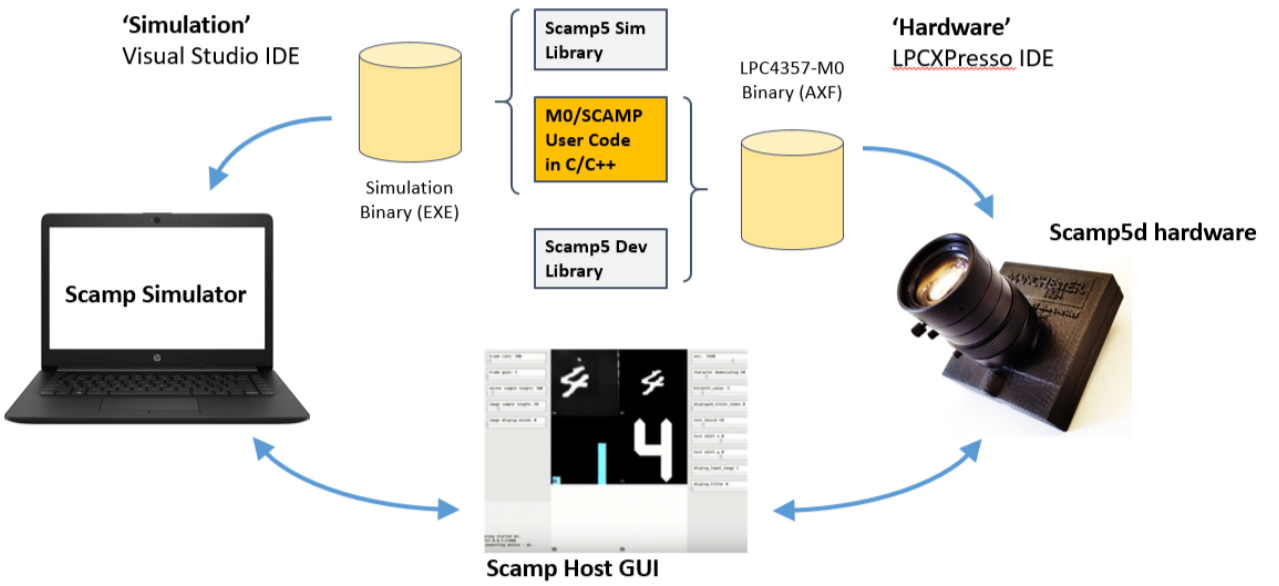

Figure 3. Scamp5d software programming framework. The same user source code can be compiled in a 'hardware' configuration to run on the Scamp5d system (right side), and in a 'Simulation' configuration, to run in emulation mode on a PC (left side).

The Scamp5d system is programmed in C/C++. The user is provided with two working environment configurations (using two separate IDEs, and two sets of libraries).

a) 'Hardware'

The user code is compiled, together with SCAMP hardware development libraries, into an executable program that runs on the M0 core. The LPCXpresso IDE (based on Eclipse) is used to develop the code. The compiled user code is programmed onto the Scamp5d system hardware over the USB (or using the flash programming tools). This configuration is available for Windows and Linux.

b) 'Simulation'

The user code is compiled, together with SCAMP simulation libraries, into an executable program that runs natively on the host PC. The Visual Studio IDE can be used to debug the code. The compiled user code is run in parallel with the **Scamp Simulator** application which provides an approximate emulation of the SCAMP-5 chip, including the ability to peek into its internal registers, step-through the code line-by-line, etc. Currently, this configuration is only available for Windows.

The Hardware configuration enables running the algorithms on the SCAMP hardware, while the Simulation configuration provides a convenient way to develop and debug the code in a PC environment. It is recommended that the environment is set up such that both Hardware and Simulation configurations are pointing to the same source code files, for a mixed development mode. (NB. It is possible to run the Simulation configuration only, for setups without hardware).

In both configurations, the running M0/SCAMP program can communicate with the host system using the scamp-interface API. Data can be send to the host (e.g. image frames resulting from the vision algorithm), and received from the host (e.g. parameters of the algorithm).

In a desktop-based development system, the **Scamp Host** application is typically used on a Windows/Linux PC to interact with the SCAMP system, and the communication is implemented over a USB interface.

It is also possible to build custom user host-side applications (for Windows, Linux, or embedded microcontrollers) using the provided host-side API. This is further explained in the Scamp5d Host-side Interface document. Embedded system development (e.g. on a robot) may require a separate configuration for development/debugging (using a PC host interface) and a separate configuration for deployment (for example, using only UART, SPI, or custom GPIO interface). The document Scamp5d as Embedded Vision System provides some guidance on this style of development.

4. System Installation

The latest version of the SCAMP software framework, installation instructions, full technical documentation of the libraries, as well as detailed user guides to the **Scamp Host** and **Scamp Simulator** applications can be found at:

https://scamp.gitlab.io/scamp5d_doc/

The ‘Simulation’ environment can be used without connecting the SCAMP hardware. SCAMP programs can be developed and simulated on a PC. This minimum configuration requires installing

- Microsoft Visual Studio IDE
- Scamp Simulation Libraries
- Scamp Simulator
- Scamp Host

The ‘Hardware’ environment requires the SCAMP hardware and installation of device drivers. It may also require microcontroller programming (Flash) hardware tools. This configuration requires installing:

- LPCXpresso IDE
- Scamp (NXP) Device Driver
- Scamp Development Libraries
- Scamp Host

Please follow Scamp5d Installation Guide to install the latest version of the SCAMP software framework.

5. Hello, world!

The SCAMP code development can be summarised as follows. The user writes a C/C++ program for the M0 processor core. The main() procedure of this program is in file typically called

"scamp5_main.cpp", but the code can be spread over several files, in the usual way. The overall code is executed on the M0 processor, but specific functions can be executed on the SCAMP-5 hardware accelerator engine.

The program under development can be compiled either for Simulation (using the Visual Studio for Windows IDE) and run on the host machine together with the Scamp Simulator, or it can be compiled for Hardware (using the LPCXpresso IDE), and run on the Scamp5d smart camera hardware. In both cases, the compiled program can interact with the Scamp Host application, which provides GUI facilities.

The code in Listing 1 illustrates a very simple SCAMP program.

```
1   #include <scamp5.hpp>
2   using namespace SCAMP5_PE;

3   int main(){
                                          Initialisation/GUI setup
4       vs_init();
5       auto display_1 = vs_gui_add_display("Hello, World!",0,0);

6       while(1){                                          frame loop
7           vs_process_message();

8           scamp5_kernel_begin();      SCAMP kernel code
9             get_image(C,D);
10            neg (A, C);
11          scamp5_kernel_end();

12          scamp5_output_image(A,display_1);

13          vs_loop_counter_inc();
14      }
15      return 0;
16  }
```

Listing 1. Hello, Scamp!

A typical SCAMP program consist of the initialisation of the system and interface (Lines 4-5), followed by the *frame loop* (Lines 6-14), which acquires the images frames, executes the vision algorithm through the combination of M0 and SCAMP-5 code, and sends the results of computations over a chosen interface (USB, UART, SPI, etc.) to a host processor or other system hardware.

The details of this are covered in the Scamp5d Programming Guide.

This example program is provided in the SCAMP software framework package as *scamp_hello_world.vprj*. Open this project in the Visual Studio IDE, compile and run. To execute the code in Simulation mode, Scamp Simulator and Scamp Host must be running first, and the Scamp Host must be "connected" to the simulator. The Scamp Host should be displaying a "negative" video from the webcam. Refer to the online Scamp5d Installation Guide for troubleshooting.

The example program is also provided as *'scamp_hello_world'* project in the default LPCXpresso workspace. Compile the program and run on the Scamp5d hardware. The Scamp Host must be "connected" to the device. The Scamp Host should be displaying a "negative" video from the Scamp5d camera. Refer to the online Scamp5d Installation Guide for troubleshooting.

6. Further reading

Scamp5d Installation Guide contains useful guides on environment setup, software updates, code repository, etc.

Scamp5d Programming Guide describes in detail the Scamp5d system and SCAMP-5 vision chip programming techniques.

Scamp5d API Reference, available online, provides detailed reference documentation for the Scamp library API

Scamp5d Simulator User Guide describes the features of the Scamp Simulator software

Scamp5d Host User Guide describes the features of the Scamp Host GUI

Code Examples included in the Scamp5d software package cover the main aspects of Scamp programming. You can use these as 'templates' for your projects.

Scamp5d Hardware Datasheet contains the details of the physical implementation, hardware specification, port pin-outs, interfaces, power supply, etc.

Scamp5d as an Embedded Vision System describes techniques used to run the system as a stand-alone device, or with a non-PC host. It also describes the methods for developing application code on the M4 core of the Scamp4d system

Scamp5d Host-side Interface describes the communication packet structure used to communicate between the Scamp5d system and the host PC. The host-side library API is also described.

References

J.Chen, S.J.Carey and P.Dudek, "Scamp5d vision system and development framework", Association for Computing Machinery, Proceedings of the 12th International Conference on Distributed Smart Cameras, ICDSC 2018, Eindhoven, September 2018

S.J.Carey, A.Lopich, D.R.W.Barr, B.Wang and P.Dudek, "A 100,000 fps Vision Sensor with Embedded 535 GOPS/W 256x256 SIMD Processor Array", VLSI Circuits Symposium 2013, Kyoto, pp.C182-C183, June 2013

Version History

Version number	Date	Author	Comments
v 1.0	03/04/2019	P.Dudek	first release
…			

