Build and install Score-P

Download and unpack Score-P. Configure it with the `--with-lib` option telling Score-P where the GASPI and IBVERBS libraries are installed:

```
$ ./configure --with-libGPI2=/home/procs/GPI2-rc2
              --with-libibverbs15=/home/procs/LibIBverb15
```

If the configuration was successful, it prints a summary to the console like this:

```
  Score-P (GASPI backend):

    C99 compiler used:          gcc -std=c99
    libibverbs15 support:       yes, using ...
    libGASPI support:           yes, using ...
    GASPI support:              yes
```

Now you can build and install Score-P:

```
$ make
$ make install
```

Setup Score-P for instrumenting

Score-P is controlled by several environment variables. Please refer to the manual of your batch system to ensure, that the environment variables are seen on each node. Two environment variables are neccessary:

```
$ export SCOREP_ENABLE_TRACING=true
$ export SCOREP_EXPERIMENT_DIRECTORY=/path/to/experiment
```

`SCOREP_ENABLE_TRACING` activates the tracing. `SCOREP_EXPERIMENT_DIRECTORY` defines the experiment directory, where the traces of your application will be stored. There are more variables to configure Score-P, please take look at the manual. For instance you can add performance counters with

```
$ export SCOREP_METRIC_PAPI=PAPI_FP_OPS,PAPI_L2_TCM
```

Instrument, compile and execute your application

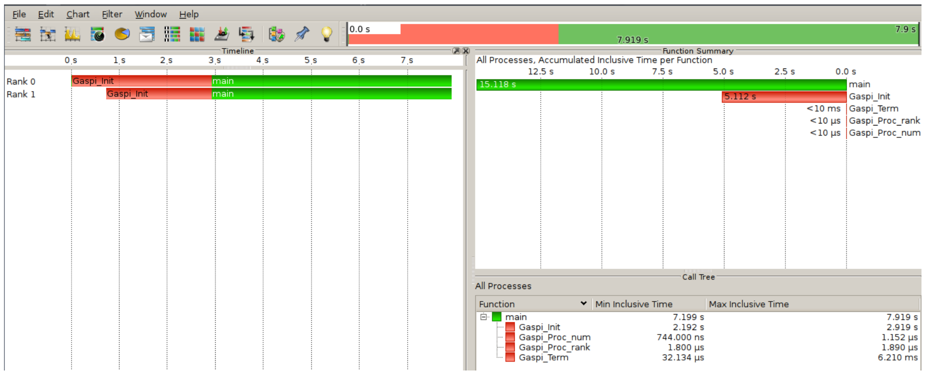
Now you can compile and instrument your GASPI application.

```
./path/to/scorep/bin/scorep --gaspi gcc ./helloworld.c
                            -o ./helloworld_scorep
```

Score-P will build an instrumented version of your GASPI application, which you can then run like the native version using `gaspi_run`:

```
gaspi_run -m ./machinefile -n 4 ./helloworld_scorep
```

After the application run has finished you can find the traces in the specified experiment directory. You can use these traces to e. g. visualize the application run using Vampir.



Function	Min Inclusive Time	Max Inclusive Time
main	7.199 s	7.919 s
Gaspi_Init	2.192 s	2.919 s
Gaspi_Proc_num	744.000 ns	1.152 µs
Gaspi_Proc_rank	1.800 µs	1.890 µs
Gaspi_Term	32.134 µs	6.210 ms

Figure 1: Vampir performance visualization