# Implementation and evaluation of an Ethernet based DAQ for a beam telescope

**Current status:**

**Profiling the software using GNU profiler**

# Basis

- **Inserted code to track the execution times**
- **Slight performance overhead**
  - Computing wise
  - Memory wise
  - Cache interference
- **Two main reports**
  - Flat profile
    - All functions and their executions times
  - Call graph
    - Execution times structured how the functions are called (tree like)

# Basis

- **Report**
  - Limited to on chip resources (size therefore time)
  - Total runtime: around 48 seconds
  - Setup:
    - Standard start up procedure
    - Connect to the two TCP Ports
    - Generate data and send it over the data connection
    - Ethernet data throughput: 2 Mbps

# Results - Flat profile

```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 28.18     13.47    13.47  1423815     0.00     0.00  memset
 16.57     21.39     7.92  1451028     0.00     0.00  memcpy
 11.23     26.76     5.37     6404     0.00     0.00  alt_busy_sleep
  7.57     30.38     3.62   794244     0.00     0.00  LOCK_NET_RESOURCE
  3.82     32.21     1.83   709330     0.00     0.00  sock_selscan
  3.18     33.73     1.52   709330     0.00     0.00  t_select
  3.05     35.19     1.46        1     1.46    36.53  SSSSimpleSocketServerTask
  2.97     36.61     1.42   603448     0.00     0.00  irq_Unmask
  2.58     37.85     1.23    22831     0.00     0.00  alt_remap_cached
  2.32     38.95     1.11   927314     0.00     0.00  OSSemPend
  1.92     39.87     0.92   905890     0.00     0.00  OSSemPost
  1.58     40.63     0.76                             alt_close
  1.57     41.38     0.75  1418661     0.00     0.00  __divsi3
  1.46     42.07     0.70   709330     0.00     0.00  bsd_select
  1.28     42.69     0.61                             loop
  1.06     43.20     0.51   794244     0.00     0.00  UNLOCK_NET_RESOURCE
  0.98     43.66     0.47  1045069     0.00     0.00  ifd_isset
  0.94     44.11     0.45    10919     0.00     0.00  tse_mac_raw_send
  0.86     44.52     0.41  1045069     0.00     0.00  ifd_set
  0.82     44.91     0.39        1     0.39     3.22  alt_tse_phy_check_link
  0.63     45.21     0.30        1     0.30     0.34  alt_main
  0.38     45.39     0.18        1     0.18     2.76  alt_tse_phy_restart_an
  0.33     45.55     0.16    16327     0.00     0.00  tcp_output
  0.30     45.69     0.14     5590     0.00     0.00  tcp_input
  0.29     45.83     0.14   416299     0.00     0.00  getq
  0.25     45.95     0.12    41205     0.00     0.00  OS_Sched
```

# Results - Call graph

```
                     Call graph (explanation follows)


granularity: each sample hit covers 32 byte(s) for 0.02% of 47.81 seconds

index % time    self  children    called     name
                                                 <spontaneous>
[1]     93.0    0.00   44.45                 OSStartTsk [1]
                1.46   35.07       1/1           SSSSimpleSocketServerTask [2]
                0.01    7.71       1/1           tk_netmain [6]
                0.00    0.08       1/1           tk_nettick [71]
                0.00    0.05    4868/4875        OSTaskStkInit [81]
                0.00    0.03       1/1           SSSInitialTask [92]
                0.01    0.01       1/1           OS_TaskIdle [105]
                0.01    0.00       1/1           OS_TaskStat [124]
                0.00    0.00       1/1           HALmain [164]
-----------------------------------------------
                1.46   35.07       1/1           OSStartTsk [1]
[2]     76.4    1.46   35.07       1         SSSSimpleSocketServerTask [2]
                1.52   28.31  709330/709330      t_select [3]
                0.00    1.80   10738/10738       sss_send_sensor_data [20]
                0.70    0.75  709330/709330      bsd_select [25]
                0.12    0.85  354664/416299      getq [38]
                0.47    0.00 1045069/1045069     ifd_isset [46]
                0.41    0.00 1045069/1045069     ifd_set [49]
                0.02    0.06   21476/64010       _free_r [58]
                0.00    0.04       1/1           sss_handle_receive_data [87]
                0.02    0.00   21476/905890      OSSemPost [40]
                0.00    0.01       1/1           sss_handle_accept_data [134]
                0.00    0.00       1/6404        usleep [12]
                0.00    0.00       2/100         printf [96]
                0.00    0.00       2/2           t_bind [181]
                0.00    0.00       2/2           t_socket [187]
                0.00    0.00       2/2           t_listen [189]
                0.00    0.00       2/3           sss_reset_connection [210]
                0.00    0.00   21476/21476       alt_uncached_free [249]
                0.00    0.00   21476/107318      __malloc_unlock [765]
                0.00    0.00   21476/21479       free [248]
```

# Results - Call graph

```
449                1.52   28.31  709330/709330      SSSSimpleSocketServerTask [2]
450 [3]     62.4   1.52   28.31  709330         t_select [3]
451               13.42    0.00 1418660/1423815     memset [4]
452                7.74    0.00 1418660/1451028     memcpy [5]
453                3.23    0.88  709330/794244      LOCK_NET_RESOURCE [14]
454                1.83    0.00  709330/709330      sock_selscan [19]
455                0.45    0.75  709330/794244      UNLOCK_NET_RESOURCE [28]
456 -----------------------------------------------
457                0.00    0.00       1/1423815     __mcount_record [161]
458                0.00    0.00       1/1423815     OS_QInit [224]
459                0.00    0.00       1/1423815     OS_MemInit [223]
460                0.00    0.00       1/1423815     OS_FlagInit [222]
461                0.00    0.00       2/1423815     ip_init [206]
462                0.00    0.00       2/1423815     alt_tse_phy_add_profile_default [200]
463                0.00    0.00       3/1423815     sss_reset_connection [210]
464                0.00    0.00       3/1423815     dhc_buildheader [204]
465                0.00    0.00       3/1423815     OSInit [108]
466                0.00    0.00       5/1423815     OSTaskCreateExt [195]
467                0.00    0.00     259/1423815     npalloc_base [145]
468                0.05    0.00    4874/1423815     OSTaskStkInit [81]
469               13.42    0.00 1418660/1423815     t_select [3]
470 [4]     28.2  13.47    0.00 1423815         memset [4]
471 -----------------------------------------------
472                0.00    0.00       1/1451028     bsd_accept [191]
473                0.00    0.00       1/1451028     t_accept [192]
474                0.00    0.00       1/1451028     ip_output [24]
475                0.00    0.00       1/1451028     tcp_input [47]
476                0.00    0.00       1/1451028     addFEBPHY [218]
477                0.00    0.00       1/1451028     alt_load [231]
478                0.00    0.00       2/1451028     t_bind [181]
479                0.00    0.00       2/1451028     alt_tse_phy_add_profile_default [200]
480                0.00    0.00       3/1451028     dhc_buildheader [204]
481                0.00    0.00       6/1451028     alt_tse_phy_add_profile [201]
482                0.00    0.00     133/1451028     altera_avalon_jtag_uart_write [167]
483                0.06    0.00   10738/1451028     sosend [22]
484                0.06    0.00   10738/1451028     tse_mac_rcv_sensor [56]
485                0.06    0.00   10740/1451028     tcp_output [18]
486                7.74    0.00 1418660/1451028     t_select [3]
487 [5]     16.6   7.92    0.00 1451028         memcpy [5]
```

# Conclusion

- **Slow execution time due to high memory demands**
- **Timeout is not the biggest factor**
- **The Select function is designed to wait for an event, which in turn reduces the memory movements**
- **Sending the data itself is compared to t_select cheap**

```
428 [2]     76.4    1.46   35.07        1          SSSSimpleSocketServerTask [2]
429                 1.52   28.31   709330/709330      t_select [3]
430                 0.00    1.80    10738/10738       sss_send_sensor_data [20]
431                 0.70    0.75   709330/709330      bsd_select [25]
432                 0.12    0.85   354664/416299      getq [38]
```

# Future work

- **Split the two connection in two threads**
  - One TCP
    - Waiting for incoming requests/data (events)
  - One UDP
    - Sending data without the TCP Overhead
- **Or even further**
  - Realize the data connection also in UDP but in Hardware
- **Some additional tweaks like an higher operation frequency or tightly coupled memory?**