CS 7639 Project 3 Part 1 Report
Crazyflie design using AADL

Part 1.1: Flow Latency analysis

Question 1 & 2:

Existing flows:

Functional model

- f_etef1 : end to end flow Gyro.f1 -> C2 -> Fusion.f2 -> C5 -> Controller.f1 -> C6 -> Motors.f1 { latency => 0 ms .. 2 ms;};
- f_etef1bis : end to end flow Gyro.f1 -> C4 -> Controller.f2 -> C6 -> Motors.f1 { latency => 0 ms .. 2 ms;};
- f_etef2 : end to end flow Acc.f1 -> C1 -> Fusion.f1 -> C5 -> Controller.f1 -> C6 -> Motors.f1 { latency => 0 ms .. 2 ms;};

System model

- etef1 : end to end flow MPU9250.f1 -> C11 -> STM32F405_Firmware.f2 -> C12 -> M1.f1 { latency => 0 ms .. 2 ms;};
- etef2 : end to end flow nRF51822_Firmware.f1 -> C10 -> STM32F405_Firmware.f1 -> C12 -> M1.f1 { latency => 0 ms .. 2 ms;};

Additional flows:

Functional model

- f_etef3 : end to end flow Magneto.f1 -> C3 -> Fusion.f3 -> C5 -> Controller.f1 -> C6 -> Motors.f1 { latency => 0 ms .. 2 ms; };
  Magnetometer is part of sensor fusion, according to crazyflie doc, that's responsible for heading and orientation of the drone. Integrating this ensures magnet data influences the motor commands
- f_etef4 : end to end flow Pilot.f1 -> C7 -> Controller.f3 -> C6 -> Motors.f1 { latency => 0 ms .. 2 ms; };
  Pilot input (human input) is directly influences the control behavior. Integrating this models the telecommand inputs which affect the motor outputs through the controller

System model

- etef1_m2 : end to end flow MPU9250.f1 -> C11 -> STM32F405_Firmware.f3 -> C13 -> M2.f1 { latency => 0 ms .. 2 ms; };
- etef1_m3 : end to end flow MPU9250.f1 -> C11 -> STM32F405_Firmware.f4 -> C14 -> M3.f1 { latency => 0 ms .. 2 ms; };
- etef1_m4 : end to end flow MPU9250.f1 -> C11 -> STM32F405_Firmware.f5 -> C15 -> M4.f1 { latency => 0 ms .. 2 ms; };

- etef2_m2 : end to end flow nRF51822_Firmware.f1 -> C10 -> STM32F405_Firmware.f6 -> C13 -> M2.f1 { latency => 0 ms .. 2 ms; };
- etef2_m3 : end to end flow nRF51822_Firmware.f1 -> C10 -> STM32F405_Firmware.f7 -> C14 -> M3.f1 { latency => 0 ms .. 2 ms; };
- etef2_m4 : end to end flow nRF51822_Firmware.f1 -> C10 -> STM32F405_Firmware.f8 -> C15 -> M4.f1 { latency => 0 ms .. 2 ms; };

Crazyflie has the quadcopter symmetrical design and original model had etef1 to M1 propeller, there are 3 other propeller it needs to take consideration of M2 M3 M4. Same thing for etef2.

- etef3 : end to end flow Magneto.f1 -> C16 -> STM32F405_Firmware.f9 -> C12 -> M1.f1;
  Routs the magnetometer data through STM32 firmware to affect the M1 motor.
  Defined a new data port "Magneto_Data" and connection "C16".
  STM32_firmware add "Magneto_Data" port and the flow "f9"
  Main_Loop in Software model defines internal flow fs9
  STM32 Firmware impl defines the connection "C10c" and full path for "f9"

In terms of configuration parameters, all the added flows have the same latency bound: "{latency => 0 ms .. 2ms;}. Sensor and control periods are set according to original time logic (4 ms for sensors, 2 ms for controller and motors)

Question 3:
After running the latency analysis, there were several end-to-end flows that exceeded their expected maximum latency of 2 ms: etef1, etef1_m2, etef1_m3, etef1_m4, etef2, etef2_m2, etef2_m3, etef2_m4, f_etef1, f_etef1bis, f_etef2, f_etef3, f_etef4.
This shows that many flows are currently too slow compared to what's required for a high performance real time control system like Crazyflie. There needs to be optimized schedules and task refinement in order to meet real time constraints.
There were high jitters, when observing the difference in minimum and maximum latency, for example: etef2_m4, jittering around 1.21 ms and 4.35 ms.

Crazyflie has a symmetrical design with four motors, which are arranged in a cross layout. Due to the design symmetry, redundant flows can be considered in the following:
Etef1, etef1_m2, etef1_m3, etef1_m4
Etef2, etef2_m2, etef2_m3, etef1_m4
F_etef1, f_etef1bis, f_etef2, f_etef3, f_etef4

Question 4:
Based on crazyflie architecture:
Sensor to Controller(250 Hz frequency > 4 ms period)
Radio to Controller(100 Hz > 10 ms)
Controller to Motors (500 Hz > 2 ms)

Affected end-to-end flows:
F_etef1, f_etef1bis = gyro to controller to motors
F_etef2 = accelerometer to controller
F_etef3 = magnetometer to controller
F_etef4 = pilot input to controller
Etef1, etef1_m2, etef1_m3, etef1_m4

Etef2, etef2_m2, etef2_m3, etef2_m4
Etef3

Most flows exceeded their maximum allowable latency, etef1 and etef2 have max latency of 2.8 ms and 4.35 ms which exceeds the 2 ms bound. F_etef1 and f_etef2 has max latency above 9.6 ms which exceeded the 4 ms. Only f_etef4 (pilot input) met the requirement, remaining within 10 ms latency.

The current architecture doesn’t meet timing requirement for most control related flows. End to end flows require additional timing allocation and thread scheduling and optimal latency to meet the system constraints

Question 5:
All thread on STM32F405 were executed. The task response times were the following:
CRTP_Rx_Task: 250 us
CRTP_Tx_Task: 300 us
Main_Loop: 200 us
Power_Management: 320 us

There were no missed deadlines and scheduling was feasible by Cheddar
The system used priority based scheduling protocol
(POSIX_1003_HIGHEST_PRIORITY_FIRST_PROTOCOL)

The number of context switches was 6 and 0 preemptions, This shows low overhead and stable scheduling pattern

Execution frequency and period aligned well with system configuration parameters

In the nrF51822, Cheddar made a warning “define a task before”. This means there were no thread defined or bound to the processor, which resulted in no execution simulated on nrF. A fix would be to assign task on nrF.

The actual behavior aligns well with the expected thread dispatch protocols and execution times
The CRTP and Power threads ran at their configured rates
The Main Loop showed periodic execution with a 200 us response time

Question 6:
The system is schedulable under the current configuration. The processor is lightly utilized, and no deadlines were violated. The results support the validity of the timing and periodic properties configured in the model.

Question 7:

Flow Analysis focuses on data latency and timing, ensuring data path meet timing constraints. In System engineering, the benefit is for requirement verification and early design validation. This way to catch early timing violations or misconfigured flows before simulation or real time deployment. Simulation focuses on dynamic behavior, shows visual representation of how task are scheduled and executed. In Systems Engineering, it would be for behavioral debugging and communication assessment. Scheduling analysis focuses on cpu-level schedulability, verifying all task met their deadlines under worst case scenarios. In System engineering, it would be for formal proof of real time feasibility. All three work together to refine the system through iterations.

Question 8:
Flow deck was added to system using AADL's refined to approach and was integrated into Hardware, Software (STM32 firmware and threads), and System (functional end to end flow etef4)

Reran flow latency analysis, scheduling feasibility and simulation in cheddar. There were no deadlines missed and all tasks are schedulable with reasonable response times.
'
The system has actual latencies between 2.8 - 4.35 ms slightly exceeding reasonable response time.s
All latencies are still within acceptable range for real time control
Flow deck adds computation time, but system remains stable and feasible