HARDWARE PLATFORM FOR RESEARCH
IN HELICOPTER UAV CONTROL

by

PETRU EMANUIEL STINGU

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON
August 2008

ACKNOWLEDGEMENTS

I would like to express my appreciation to my supervisor, Dr. Frank Lewis. His research interests in adaptive and robust control, as well as in the coordinated control of multiple ground and aerial vehicles motivated me to design and build the hardware that can later implement some of those theoretical results. I especially like to thank him for giving me the freedom to choose the subject and the ideas for my research and for supporting my work.

I am also very grateful to Dr. Kamesh Subbarao and to Dr. Jason Losh. Their courses were some of the best that I have attended and provided me with the foundation on which to develop my current and future projects.

The support of the Army Research Office and Army National Automotive Center in grant W91NF-05-1-0314 is gratefully acknowledged.

July 10, 2008

ABSTRACT

HARDWARE PLATFORM FOR RESEARCH IN HELICOPTER UAV CONTROL

Petru Emanuiel Stingu, M.S.

The University of Texas at Arlington, 2008

Supervising Professor: Frank L. Lewis

The topic discussed in this work addresses the current research being conducted at the Automation & Robotics Research Institute in the areas of UAV helicopter instrumentation targeted towards aerobatic flight control and heterogenous multi-vehicle cooperation. A modular electronic control system is designed which includes sensors communicating over a CAN bus and an on-board computer that runs a special real-time Linux kernel. The majority the electronics is developed by the author, including the electronic schematics, PCB design, component soldering and the supporting software. Commercially available modules are used for the radio transceivers or the on-board computer. The same control system is reusable on other types of vehicles and on the ground stations, facilitating the development of multi-vehicle control algorithms. The control algorithms are developed in Simulink, converted to C code and compiled into a special application that runs on the on-board computer or on the base stations. Special blocks are developed for the sensors and the actuators, but also for entire vehicles, providing very easy access to the process data in Simulink.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

During recent years, the contributions of unmanned systems to military and civilian operations continue to increase, reducing the risk to human life. Unmanned Aircraft Systems (UASs) can play an important role in missions such as reconnaissance and surveillance, precision target location, signals intelligence, and digital mapping. Because of the miniaturization of sensors and communication equipment, smaller aircraft can now perform the same mission that would have required a bigger aircraft and in some cases a person a few years ago. This has obvious advantages in terms of costs and logistics, but also it enables completely new missions, such as operation at street level in urban environments. UAV helicopters are particularly useful because of their ability to hover and to take off vertically.

Many research efforts have been made towards the control of helicopter UAVs [1-4]. Most of them use linear models for the helicopter dynamics and are concerned mostly with hovering, forward flight or simple maneuvers. The exceptional capabilities of these small-scale helicopters to do aerobatic flight maneuvers [5] due to the high thrust to weight ratio (that can reach 3) have not been completely explored. Even more, most of the platforms used to implement the algorithms were modified in order to add instrumentation and on-board computers. The common approach was to include most of the additional components in a heavy box under the helicopter body, by replacing the original skids with taller skids that could accommodate the equipment height. The extra weight placed well below the initial center of gravity enhances the stability of the helicopter, but severely limits the aerobatic flight capabilities.

Most of the research platforms in helicopter UAV control are owned by universities: Carnegie Mellon [6], Massachusetts Institute of Technology [5], Georgia Tech [7, 8], National

University of Singapore [9], Berkeley [10], University of South Florida [2], and many others. There also exist several commercial vendors of small-scale UAV helicopters, the most important being Neural Robotics Inc. and Rotomotion. Although there are publications that describe the hardware implementation and the control algorithms used for most of these platforms, it is practically impossible for anyone to duplicate their implementations due to the lack of details in the information that is released to the public. This is especially true for the implementation of the electronic system and the supporting software, the information about the control algorithms being generally rich enough. Another obstacle in using an already available platform as a reference is the fact that many of them use old electronic modules that are not available any more. There is no reason to start building a helicopter using old hardware when there are better solutions on the market. The third obstacle is the mechanical helicopter platform itself. Most projects use the Yamaha RMAX platform, which is an industrial scale remote controlled helicopter. It is too big and expensive and is not suitable for a small research group. Even more, it is currently restricted for export from Japan and can not be purchased in the United States. The simplest solution appears to be the design of a completely new UAV helicopter platform that can take advantage of the current technologies in computing, communications and other fields and where full control will be maintained over the hardware and the software aspects.

This work addresses the current research being conducted at the Automation & Robotics Research Institute in the areas of UAV helicopter instrumentation targeted towards aerobatic flight and heterogenous multi-vehicle cooperation. A slightly different approach is offered to what has already been done in this field, by proposing a modularized system that can be configured to work on different types of vehicles with minimal effort. The following chapters describe the control system for one vehicle and the overall structure of the system that allows coordinated control for multiple heterogenous vehicles. More details are given on the hardware implementation of the sensors, the real-time controllers and the on-board computers, but some aspects of the software are also covered, especially those related to real-time control.

1.1. Problem Description

There are a few main ideas that have guided the design of the control system. *Reusability* is important in order to reduce the development time and costs. The same electronic modules and the same software can be used on the helicopters, on the ground vehicles and on the ground monitoring stations. The system is *modular*, consisting of discrete electronic modules that communicate over a CAN bus. This architecture allows the system to be *distributed* across the whole helicopter body (Fig.1), having a small effect on the center of gravity and allowing aerobatic maneuvers.

Safety is a big concern because helicopters of this size are expensive and present the potential of being very dangerous to the operator. In order to increase the robustness of the electronic control system, the computing functions are handled by both the onboard Pentium M processor and by a microcontroller on the Real-Time module that can take control when the former fails. The microcontroller has access to an additional set of local (low precision) inertial sensors so it has more chances of safely landing the helicopter when something goes wrong with the software or with the electronics. The CAN protocol is relatively insensitive to electromagnetic interference and assures a reliable communication link between the modules. The power supplies for different functions are separated and short-circuit protected. In case of a wiring failure, the remainder of the system will still be operational. Various current, voltage, temperature and speed sensors allow prediction of failures before they get critical.

Low power consumption and small weight are very important for an UAV, because they determine its flight time and the maneuverability. The specially-designed electronics are smaller and weight less than the corresponding off the shelf components. All the power conversion from the battery to the various modules is done using high-efficiency switched-mode power supplies, thus optimizing the use of electric energy and allowing longer flight times.

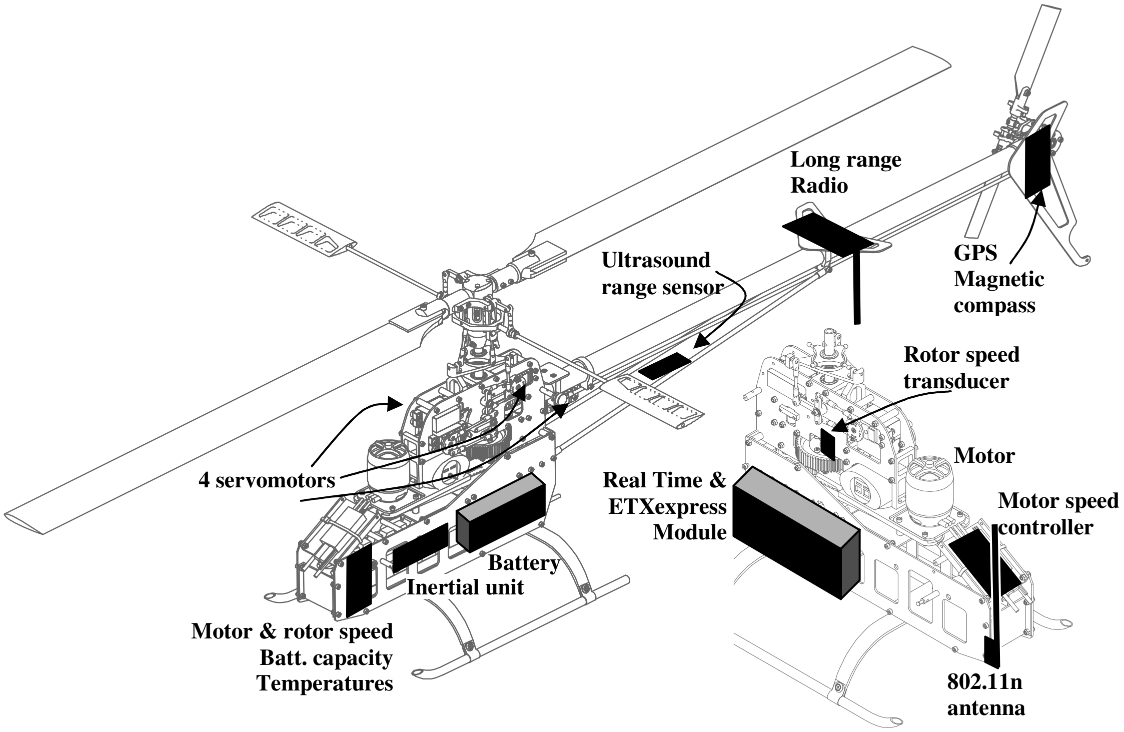


Figure 1. Placement of the electronic modules on the helicopter body

The mechanical platform is a Joker 2 fully aerobatic electric helicopter (Fig. 2), manufactured by Minicopter Germany. The main technical details are the following:

- Main rotor diameter: 1560 mm
- Tail rotor diameter: 290 mm
- Weight: 4.5-5.0 kg (with batteries)
- Maximum total weight: 6.8 kg
- Flight time: 8-20 min depending on flight style and load
- Batteries: two Lithium-Polymer, 8000 mAh, 18.5 V
- Motor: Plettenberg 370/40/A2/Heli, 3-phase brushless motor, 2.5KW, low vibrations.

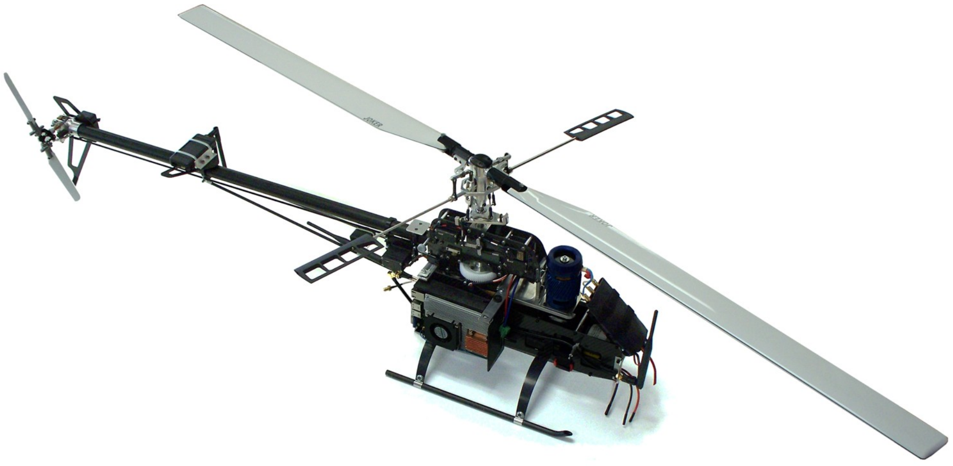


Figure 2. The Joker 2 helicopter platform

The main rotor has a Bell-Hiller stabilizer bar, but it can be replaced with a rigid rotor head with no stabilizer bar at all, for improved 3D aerobatic maneuverability. The choice of an electric helicopter instead of one powered by a combustion engine was due to system identification issues because of the fuel consumption and the change of the helicopter mass, which makes it impossible to obtain consistent measurements at every experiment. It is also easier to work with electric motors instead of combustion engines that have to be periodically tuned and have to be started manually.

Some of the experiments are done with the helicopter tied to a specially designed helicopter stand (Fig. 3) that allows the helicopter to tilt in any direction (within some limits), to rotate around the vertical axis, to go up and down and to rotate in a circle in the horizontal plane. The weight of the stand is cancelled using an air cylinder. The stand is most useful to trim the helicopter and to obtain the linear model for hovering.

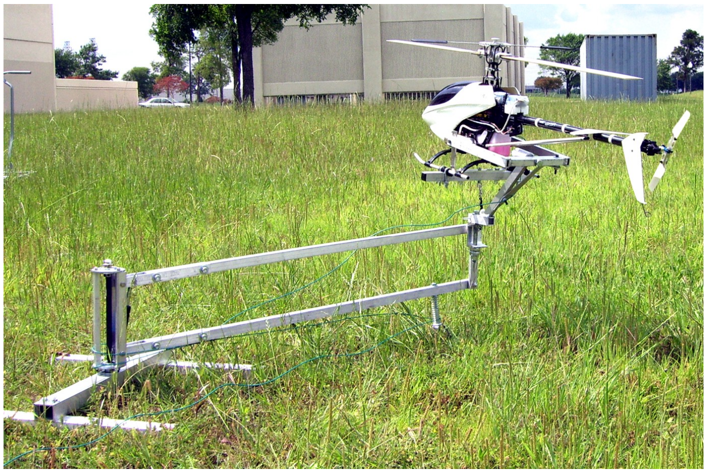


Figure 3. The helicopter stand with a Raptor 30 V2 helicopter

CHAPTER 2

ORGANIZATION OF THE HELICOPTER CONTROL SYSTEM

The electronic control system is distributed in three locations: on the helicopter body, on the ground (base station) and at the human pilot for emergency situations (remote control). Because the onboard computer is sufficient for control, the ground base station is optional. The radio communication is realized using two distinct transmit/receive links. The first one, in the 900MHz band, uses high-power transceivers to cover a radius of about 14 miles with simple omnidirectional antennas. The shortcoming of this connection is the fact that it only works at low speeds, having a maximum bit rate of 115.2 Kbps. At this speed, the helicopters can transmit to the ground monitoring station only the most important flight variables and can receive commands from the ground station and from remote controls in case of emergency. The second radio link is a high-speed 802.11n wireless network, having a maximum throughput of 300Mbps. The range is about 300 meters. The network uses UDP packets and allows each device to communicate with any other. This allows the implementation of formation flight algorithms, because the helicopters, the ground vehicles and the ground stations can easily share information in real-time. If the high-speed connection is lost, the long-range link is used to maintain control and make the vehicles return inside the area covered by the high-speed radio link.

2.1. Overview of the On-Board System

The main idea that has driven the design of the on-board electronic system was the ability to spread the components across the helicopter body in such a way that the center of gravity does not shift considerably compared with the original platform. The approach of having a large box containing all the electronic system under the helicopter body was inacceptable because the center of gravity would have moved down too much and would have made the helicopter more stable, preventing aerobatic maneuvers. This issue only exists for small-scale helicopters.

Because the system components are spread apart, a reliable communication channel had to be adopted. A factor that puts more pressure on the choice of communication protocol is the fact that the helicopter is electric. The motor can have current spikes of 100A. The average current consumption is of 25A for hovering. The speed controller for the brushless motor has to switch the power to different windings synchronously to the rotor movement and controls the power delivered to the motor by PWM. All this high-current and high-voltage switching causes additional electromagnetic interference that can potentially disrupt the communication between the modules of the control system. The long-range radio transceiver has a maximum transmitting power of 1W. Some of the energy transmitted can be induced into the wiring on the helicopter body and rectified inside the semiconductor components, causing additional communication errors. The solution to all these problems was to choose CAN as the communication standard. The Controller Area Network (CAN) is a high-integrity serial data communications bus for real-time control applications originally developed by Bosch for use in cars. It has very good error detection and confinement capabilities, which coupled with the differential signaling ensure a high degree of robustness. It allows real-time communication due to the following characteristics: relatively high speed (1 Mbps), small latency due to short packets, and node addressing with bitwise arbitration that prevents collisions and allows prioritization of messages.

The block diagram of the on-board electronic system is shown in Fig. 4.

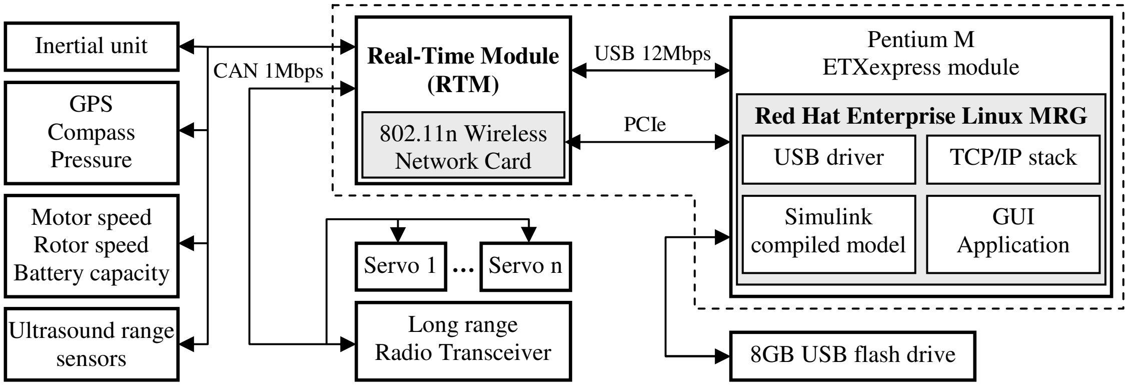


Figure 4. Organization of the on-board helicopter control system

The most important component is the Real-Time Module (RTM). It acts as a communication bridge between the CAN bus and the USB bus (Fig. 9), provides power to all the modules and implements some simple control algorithms that are activated in emergency situations. It is also a carrier board for the Pentium M computer module that actually runs the complex control algorithms (in normal conditions). The computer module runs Red Hat Enterprise Linux with a real-time kernel. It connects through the PCI express bus to a 802.11n wireless network adapter installed on the RTM and also communicates with the RTM using a 12 Mbps USB link. The two modules are assembled in a stack and placed in an aluminum box on one side of the helicopter (Fig. 5). The CAN bus is used to interface sensors and actuators to the RTM. The USB link between the RTM and the computer module allows the transfer of sensor data towards the computer at the beginning of each sample period and the transfer of actuator commands from the computer to the RTM at the end of each sample period.

Each sensor or group of sensors is contained in a separate plastic box and communicates with the Real-Time Module via a CAN bus at a data rate of 1Mbps. In addition to receiving commands, the servomotors can also send their position back to the RTM.

Most of the CAN-enabled devices are built around a Microchip PIC18Fx580 CAN microcontroller. It is the interface between the nonstandard peripheral and the standardized CANopen protocol used to communicate with the rest of the system (Fig. 6). All the particularities related to the implementation of the device are handled locally by the microcontroller firmware. The custom functionality varies from just a bridge between the CAN and the RS-232 protocols for the long-range radio transceiver to more complex time-sensitive measurements and supervising functions in the case of the system monitoring module.

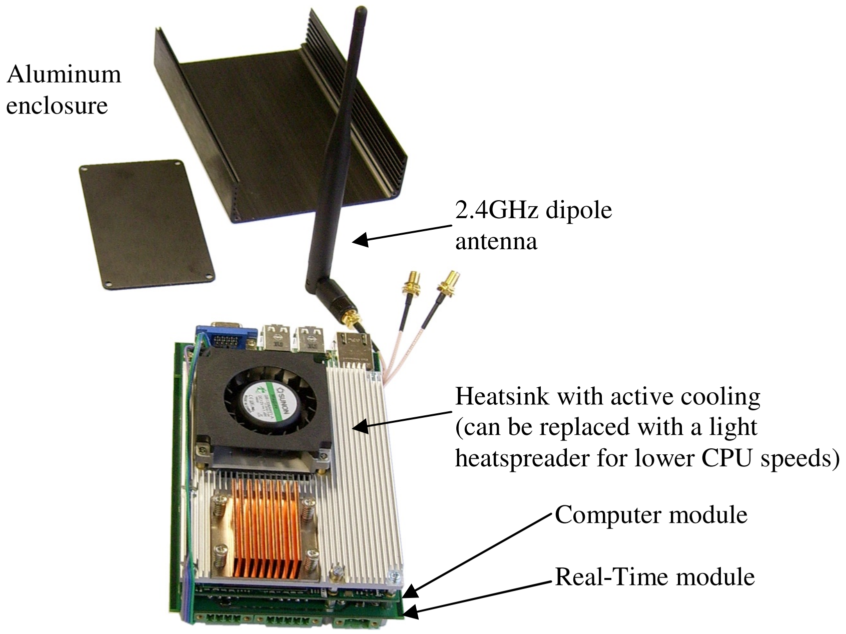

Aluminum enclosure
2.4GHz dipole antenna
Heatsink with active cooling (can be replaced with a light heatspreader for lower CPU speeds)
Computer module
Real-Time module

Figure 5. The Real-Time and Computer Module stack

The majority of the modules on the CAN bus receive their power supply from the RTM. The 9V supply is locally converted into the necessary voltages using either switched-mode or linear voltage regulators, depending on the power consumption of each module. This way, disturbances present on the external power supply wires do not affect the correct functionality of the sensitive electronics inside them.

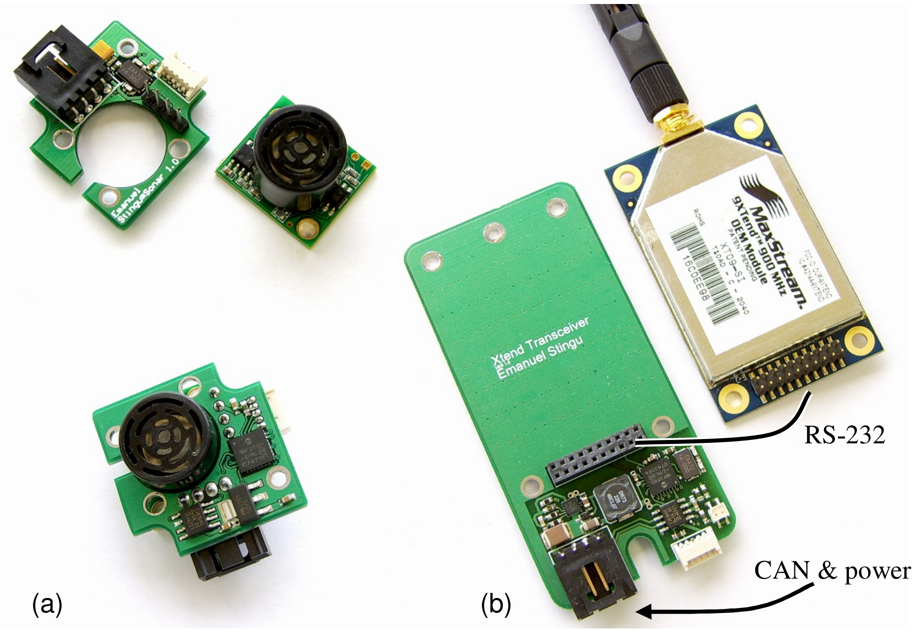


Figure 6. Examples of interfacing some devices to the CAN bus. A MaxBotix ultrasound range finder (a) and the XTend long-range radio transceiver (b)

2.2. Computer Module

Before the decision was made on what processing platform to use, a review of the existing technologies was made. The goal was to build a cheap system where one can have complete control over the software and also be able to easily extend the functionality when needed. Besides that, the ease of implementation of the control algorithms was also considered.

There are various possible choices for the architecture of an on-board computer. DSP and FPGA platforms are attractive due to the high degree of parallelism, the low power consumption or the good floating-point performance. Unfortunately, all the benefits vanish when it comes to actually implement a complete system. The software components that allow interfacing to USB, Ethernet and PCI express as well as the drivers or libraries necessary to communicate with peripherals like USB video cameras and wireless network cards are not open-source or free (in general). Because the helicopter is a research platform, most of the algorithms will not be optimized for speed or for parallelism. They will need strong floating-point performance and might not run optimally on fixed-point DSPs. FPGA implementations require the conversion of the C or Matlab code into VHDL or Verilog code, then testing and synthesizing, a process that requires specialized knowledge in FPGA design.

Due to the reasons stated above, the choice for the architecture of the on-board computer was to use an x86 general-purpose processor that has the largest amount of software support and adapts reasonably well to any type of control algorithm. The best performance per watt ratio (especially for floating-point tasks) at the time of system design was reached by the Intel Pentium M processors (Dothan) and by the Intel Core 2 Duo processors (for parallel algorithms).

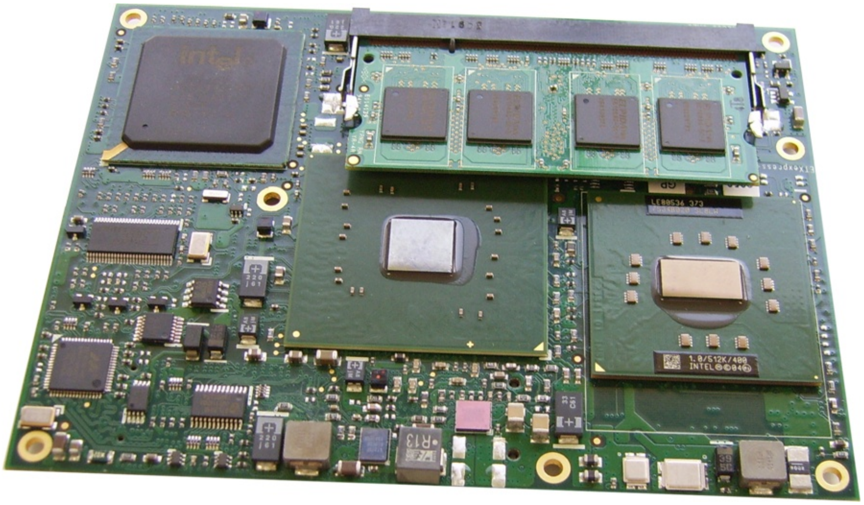


Figure 7. ETXexpress module manufactured by Adlink Technology Inc.

The smallest form factor for the CPU board that contained the desired processors and all the necessary peripherals was the COM Express computer-on-module (also named ETXexpress). It is a highly integrated off-the-shelf building block based on a PCI Express bus architecture that plugs into custom made, application-specific carrier boards (in our case, the Real-Time Module). COM Express modules (Fig. 7) measure just 95 mm x 125 mm and include generic functions such as video, audio, Ethernet, storage interfaces and USB ports that are needed for most applications. The components present on the modules are the processor, the chipset (with on-board graphics), the DDR memory, one Gigabit Ethernet controller and the necessary switched-mode power supplies that convert the input 12V voltage to the necessary levels needed by the different circuits. Due to their small size, the modules do not have specialized connectors for the various peripherals. Instead, they connect through a pair of high-density connectors to the carrier board, which routes the necessary signals to their connectors if needed. The RTM makes available 5 USB ports, one Gigabit Ethernet port and a combined analog video and TV output. It also routes one PCI express lane to a mini PCI express wireless adapter and one USB channel to the RTM microcontroller. There is a small number of other signals used, mainly to allow the RTM microcontroller to do power management and supervise the behavior of the ETXexpress module. Most of the signals available from the ETXexpress module are not used. Especially important are the unused PCI express lanes that can be routed to a FPGA on the RTM PCB in a future design.

The hardware configuration of the ETXexpress modules is compatible with Microsoft Windows and Linux, so they can be used the same way as normal computers, without the need of customized software.

2.3. Real-Time Module

The Real-Time Module (Figure 8) is built around a 32-bit ARM microcontroller running at 48 MHz. The main role of the microcontroller is to support the communication on the CAN bus and on the USB bus. It has a lightweight free real-time operating system (called exactly like that: FreeRTOS) that provides the basic functionality expected from a RTOS: tasks, preemptive and cooperative scheduling, semaphores, mutexes, queues, critical sections and many others. Special drivers were written to support low-latency communication for the USB bus, the CAN bus and the RS-232 serial ports (Figure 9). They use circular buffers and take advantage of the

DMA transfer capabilities of the microcontroller, allowing the hardware to do most of the work and guaranteeing the fastest possible response to communication events without requiring software intervention.

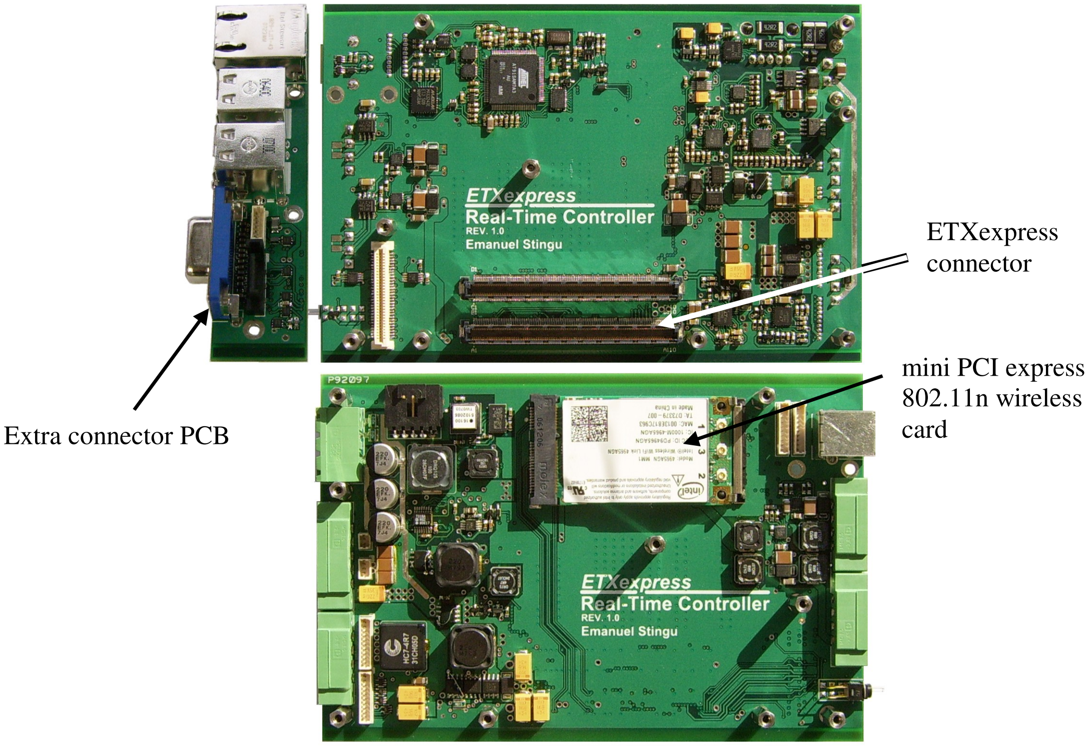

ETXexpress connector
mini PCI express 802.11n wireless card
Extra connector PCB
ETXexpress Real-Time Controller REV. 1.0 Emanuel Stingu

Figure 8. The Real-Time Module

The microcontroller connects via USB with the ETXexpress computer module or to an external computer. The channel is used twice during a sample period: at the beginning, the microcontroller sends the sensor data to the computer module, and at the end of the sample period it receives the commands to be sent to the actuators. The time taken by the USB communication has to be minimized and that is why all efforts were done to optimize the drivers for low latency in order to use the USB bus at its full capacity. The USB connection is also used by a custom bootloader that allows the firmware to be updated easily.

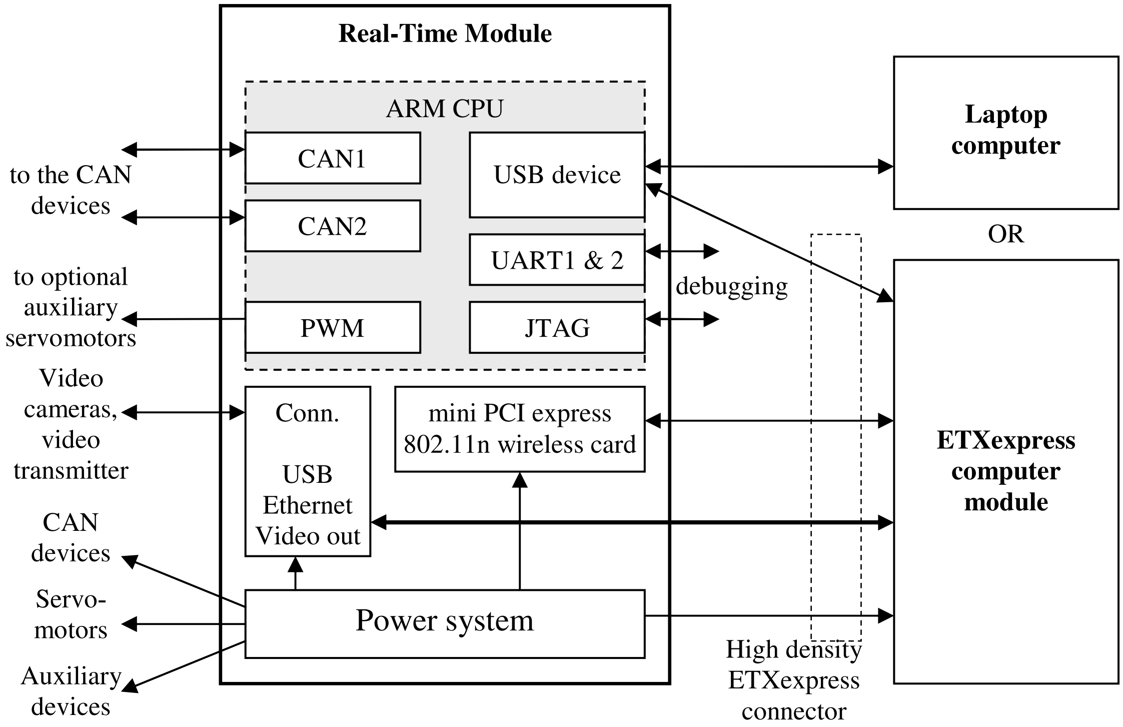


Figure 9. Connectivity of the Real-Time Module

The RTM provides two CAN channels for connecting external sensors, actuators and special function devices. The communication uses CANopen [11], a CAN-based higher layer protocol. It provides standardized communication objects for real-time data, configuration data as well as network management data. Each device on the CAN bus has its own identifier and an associated Object Dictionary (OD) that provides a complete description of the device and its network behavior. All messages are broadcast on the CAN bus. Based on the entries in the OD, each device implements a filter that allows the acceptance of only the messages of interest. The following predefined types of messages exist on the network:

- Administrative messages

  They are based on a master-slave concept. On a CAN bus there is only one master (for each specific service) and one or multiple slaves. The following services are implemented:

– *Synchronization* - used to synchronize all the devices to the sample time. The RTM is the master. The CAN identifier associated to this message has the maximum priority and always wins the bitwise arbitration with all the other messages that are ready to be sent at the same time. Because each CAN node has a hardware timer that timestamps the CAN messages, the synchronization is precise to 1 microsecond.

– *Boot-up* - the slaves have four possible states: Initializing, Pre-operational, Operational and Stopped. The slaves send this message to indicate to the master that they have transitioned from state Initializing to state Pre-operational. The RTM sends this message to start, stop or reset one or all the slaves.

- Service Data Objects

  These are messages that access the Object Dictionary of the slave devices. The master (the RTM) can thus interrogate each slave and determine the supported network services and the available real-time variables and can update its configuration. While the slaves are in the Pre-operational state, the master configures the sample rate, the time offset of different actions relative to the SYNC message and other specific parameters. When the configuration is done for all the devices on the CAN bus, the master transitions them to the Operational state. In order to allow updating of the firmware for the CAN devices without removing them from the system, a CAN bootloader was implemented. It is active in the Pre-operational state and accessible using SDO messages. The RTM receives the CAN identifier for the device that has to be programmed and the HEX file over the USB bus from the ETXexpress computer module or from an external computer and sends it via the CAN bus to the device to be programmed.

- Process Data Objects

  These are messages that encapsulate the contents of real-time variables that are sent or received by the CAN devices during each sample period. They are based on a producer-consumer concept. Data is sent from one (and only one) producer to one or multiple consumers. In normal conditions, the RTM is the consumer for the data produced by the sensors, and the actuators are the consumers for the commands sent by the RTM. In the case where the RTM stops responding, the CAN devices change behavior and implement a distributed control algorithm that tries to stabilize the helicopter and land. In order to accomplish this, the actuators become consumers for the sensor data.

The USB interface implements the Communication Device Class (CDC) that appears as a virtual COM port to the USB host, which greatly simplifies the application programming. It uses three endpoints. The default control endpoint 0 is used for device management. The other two are a pair of IN and OUT endpoints that can use Bulk or Isochronous transfers. In the current implementation, Bulk transfers are used because they guarantee the correct delivery of data and no significant latency was observed when only one device is connected to the USB root hub. In the future, to improve the real-time behavior, isochronous transfers will be used because they guarantee a minimum latency. The retransmission of data in case of errors will be handled in software. The USB communication uses two buffers for each data endpoint. While the software handles the contents of one buffer, the hardware works with the other. This way, the hardware interface does not generally have to wait for the software to fill the buffers, allowing the full bandwidth of USB to be used.

The Real-Time Module also contains 10 switched-mode power supplies to power all the different modules installed on the helicopter (see Fig. 10). All are current limited and short-circuit protected and automatically recover from an over-current condition. All the power supplies can be individually enabled by the microcontroller and all report over-current conditions for easy localization of defects in the system. A high-current switched-mode Li-Po battery charger was included to eliminate the need of removing the battery for external charging. It also supplies the internal power bus when the DC adapter is connected. It gives priority to the internal devices and only the remaining current is used to charge the battery. The cell balancer is needed to guarantee that all cells charge equally even after a high number of charge/discharge cycles.

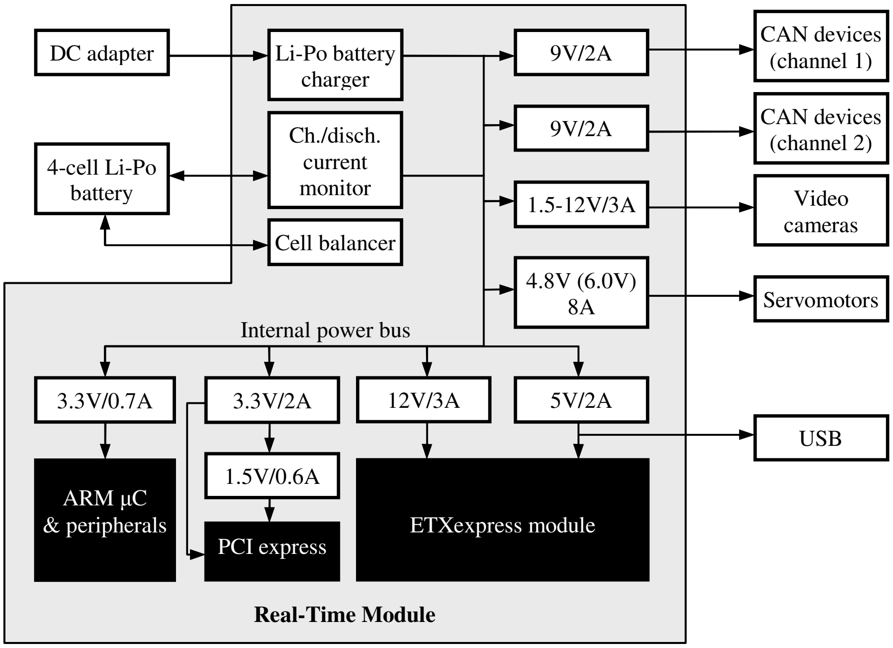


Figure 10. The power system on the Real-Time Module

The charge and discharge current is permanently monitored by the microcontroller in order to create the function of a “fuel gauge” that will inform the pilot and the control algorithm of the available flight time for the on-board electronics. Each CAN channel is powered by its own power supply for increased reliability. A short-circuit on the power lines of one channel will not disable all the functionality of the external system. The 1.5V-12V programmable power supply was included to allow powering video cameras for stereo vision and any other device that might need to be added in the future.

2.4. Inertial Measurement Unit

The Inertial Measurement Unit (IMU) is one of the most important components of the control system. Its measurements are critical for estimating the full state of the helicopter during flight when the other measurements for absolute position and orientation are absent or imprecise. All efforts were made to reach the maximum possible accuracy of this instrument using relatively inexpensive sensors. It uses MEMS accelerometers and gyroscopes with high internal resonant frequency in order to insure reliable vibration rejection over a wide frequency range and in order to have a higher bandwidth.

Most of the commercial solutions use a single A/D converter with a multiplexer to convert all the six analog channels to digital representation. Each sensor is sampled at a different instant and the data set provided for all the sensors is not consistent if the sample rate is low. Another issue is the bias and sensitivity calibration with temperature, where the usual 3-point calibration technique does not offer sufficient precision.

Because the cost of analog to digital converters is much lower than the cost of the inertial sensors, the solution to all of the above problems was to use a dedicated sigma-delta A/D converter for each channel. This approach has the following advantages:

- Provides consistency of the sampled signals (all are sampled simultaneously).
- Allows faster sample rates because the converter time is fully used by one channel and multiplexer settling times are eliminated.
- Avoids the use of complex signal conditioning circuitry that can cause bias, sensitivity errors and temperature drift. This can be achieved because there is no need to buffer the sensor outputs as there is no multiplexer present any more and the input of the A/D converter allows the signal to completely settle after each sample.
- Only a simple RC filter is needed for anti-aliasing purposes because the sampling frequency of the sigma-delta converters is about 450 times faster than the data rate of the A/D converter.

Even more, a custom solution allows for setting the sample points at well-defined moments in time, synchronized with the rest of the sensors available on the platform.

For temperature compensation, the same type of A/D converter is used as for the inertial sensors to avoid introducing additional errors to the final results by using imprecise or noisy temperature data. There are about 100 points used for temperature calibration. For all these temperature points the raw measurements for all the sensors were collected. They include the bias values obtained with the IMU being stationary and sensitivity data obtained using the gravity for the accelerometers and a constant-speed rotating platform for the gyroscopes. More complex experiments will be designed in order to enhance the calibration performance and take into account the cross-axis effects and the alignment errors.

Another aspect that had to be addressed especially in the case of the helicopter platform was to find a way to eliminate the vibration effects from the output of the IMU. Using electric helicopters instead of gas-powered ensures that there are less vibrations, but this is not enough. Using mechanical vibration dampers between the helicopter body and the IMU deteriorates the precision of the measurements by adding biases, phase delay and some nonlinearities, all of these being temperature dependent. Using analog filters at each sensor output complicates the design too much and doesn't allow for adaptability. The problem was solved using a combination of solutions (Fig. 11). First of all, the vibrations with a frequency higher or close to the internal resonant frequency of the sensor were filtered using a mechanical damper with very good characteristics but also very simple: dual-sided adhesive tape of a certain height. A RC filter was used to lower the bandwidth of the sensors to about 2 KHz. This limits the demodulation noise inherent to this kind of resonant sensors. The sigma-delta A/D converters sample the signal at 1.8MHz and provide output data at a rate of 4 KHz. The IMU is driven by a 32-bit ARM microcontroller that applies a FIR filter that down-samples the data to a rate programmed on the CAN bus (currently 100 Hz). The filter coefficients are also programmed over the CAN bus during the initialization of the sensors.

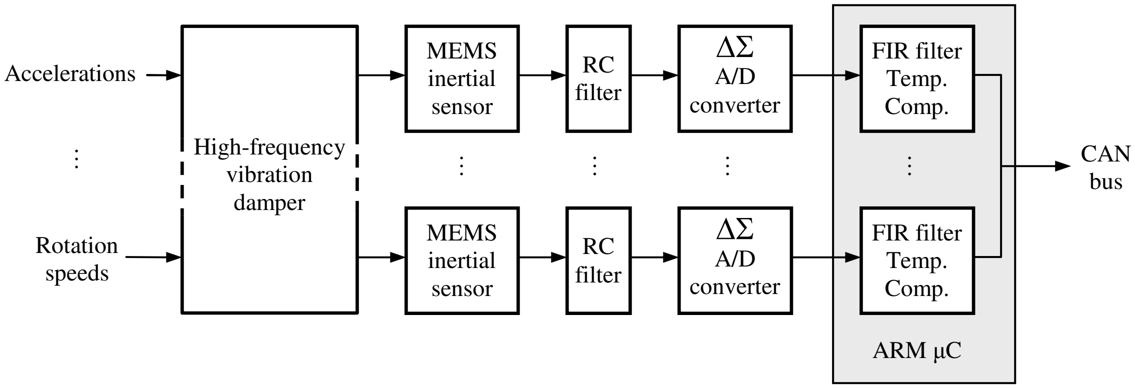


Figure 11. The signal paths inside the IMU

2.5. System Monitor Module (SMM)

The safety of the helicopter and of the human operators has to be enforced using all possible means. The operation of an electric helicopter of this size requires the strict observation of some specific parameters during flight. The System Monitor Module is the interface between the low-power electronic system (that contains the on-board computer and most of the sensors, implementing the control algorithm) and the high-power electro-mechanical system that actually makes the helicopter fly. The two electronic/electric systems have separate batteries in order to prevent interferences from the high-current and high-voltage system going into the sensitive low-voltage system. Because the SMM monitors most of the parameters in the power supply domain of the helicopter motor, it receives its power from the main Li-Po batteries. The CAN bus is placed in the power domain of the on-board computer. To allow communication over the CAN bus, the SMM uses optical isolation between its microcontroller and the CAN transceivers.

The SMM measures the following parameters (Fig. 12): the voltage for each of the two main batteries, the current through the main batteries, their temperature, the motor temperature, the motor speed and the main rotor speed. It also provides the command to the motor speed controller (as a servo-type PWM signal).

Lithium-Polymer cells are not allowed to reach a voltage lower than 3.0V. If they do, irreversible reactions take place that can severely limit the battery capacity for subsequent charge cycles or even worse they can cause fire. Therefore a critical function of the SMM is to continuously monitor the voltage for each battery. In the case where the control system does not react and land the helicopter, the SMM slowly lowers the RPM of the motor to do a forced landing by itself. The current monitoring function, together with the voltage monitoring, allows measurement of the instantaneous power required by the helicopter. This is used to implement a “fuel gauge” that indicates the remaining battery time, but can also be used in the control algorithms to optimize for power consumption.

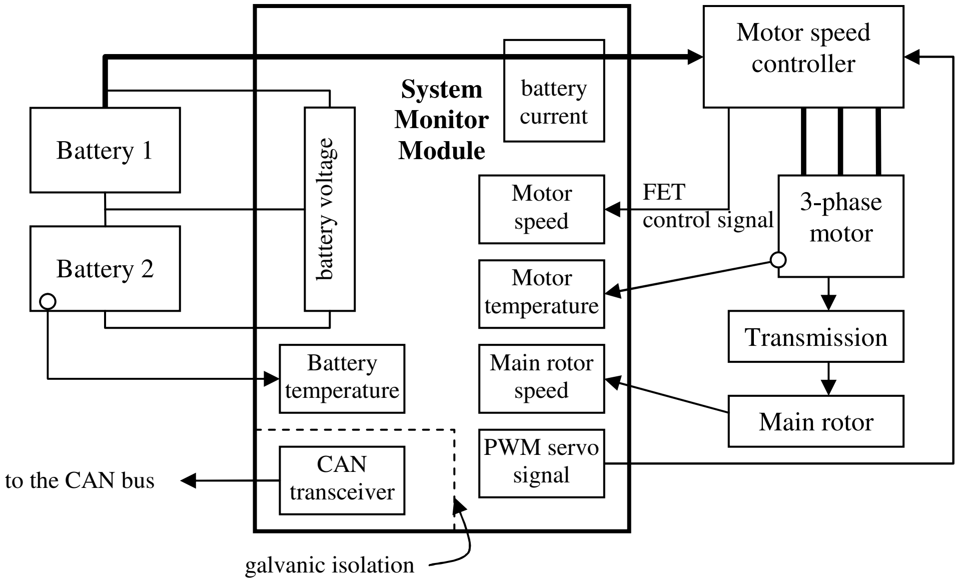


Figure 12. The System Monitor Module

The motor is a three-phase brushless motor with no sensors to measure the shaft angle or speed. The Electronic Speed Controller uses PWM power signals to control the speed. It connects two phases of the motor at one time to the battery and measures the third to get the back-EMF voltage generated by the motor while it rotates. The zero-crossings of this measured voltage are used to detect the angle of the rotor and to switch the power to the next pair of phases. Because installing speed sensors on the motor is not a viable solution, the measurement is done on the command given by the speed controller to the low-side FET transistors that switch one of the phases. The phase-switching is synchronous with the rotor movement therefore the exact motor speed can be detected. The speed of the main rotor of the helicopter is sensed using an integrated Hall sensor. The two speeds are not necessarily an expression of the gear ratio. The motor drives the main rotor through a timing belt and a gearbox. Further, there is a one-way bearing installed that allows the main rotor to freewheel. One more case when the two speeds do not follow the gear ratio is when there is a mechanical failure and the timing belt skips teeth. In this situation, the main rotor speed is smaller than the minimum expected from the gear ratio and the motor speed. This case is continuously monitored and signaled to the control algorithm and to the human operator in order to land the helicopter before the problem gets worse.

2.6. Servomotors

The nonlinear model of the helicopter can be written in the most general form

$$\dot{x} = f(x,u).$$

The u vector can have multiple components, but four of them are the different displacements of the servomotors that control the angles of the blades:

δ_{front}, δ_{left}, δ_{right} - determine the swashplate position

δ_{tail} - determines the angle of the tail rotor blades

Usually servomotors receive the reference angle as an input and an internal feedback loop rotates the servomotor disc to that position. In steady-state, the true position of the servomotor will be equal to the reference command. Because the reference is known, it is used in many similar projects to estimate the real position of the servomotor (which is not available for measurement) by approximating the dynamics of the servomotor with a time delay or a better model (Fig. 13).

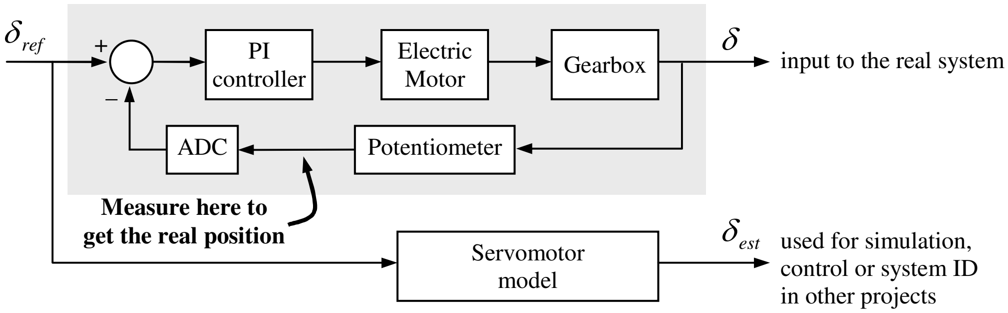


Figure 13. Servomotor block diagram

The problem with this approach is the fact that in most situations the servomotor model is not known, and even if it is known, the dynamics depends on a few parameters like external forces or the supply voltage. The solution is to measure the voltage on the potentiometer installed inside the servomotor that provides the position feedback to the internal control loop. For each servomotor, an extra PCB was added to the existing electronics in order to interface it to the CAN bus (Fig. 14). The new electronic module reads the potentiometer voltage and generates the PWM signal that is used by the servomotor to calculate the reference position. This way, the true position of the servomotors is available to the parameter estimation algorithms in order to achieve better precision.

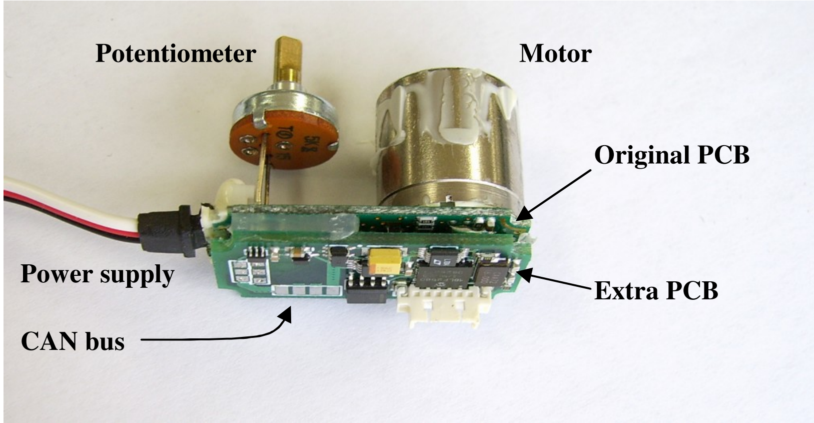


Figure 14. Extra PCB used to measure the position of the servomotor

2.7. Radio Transceivers

The helicopter communicates using two radio interfaces. For safety reasons it uses a long-range (14 miles) radio link on 900 MHz such that it is always able to receive commands from the remote control and can communicate with the optional base station. The high-speed link is a 802.11n wireless network connection, used to communicate with all the other devices (helicopters, ground vehicles, base stations).

The long-range radio transceiver is a module available on the CAN bus (Fig. 6). It includes an XTend RF module manufactured by Digi International. The RF data rate is 125 Kbps. At this relatively low speed, only the most important states of the helicopter can be sent in real-time at a rate of 100 Hz. The complete information is sent using the 802.11n wireless network. There are three devices present on the 900 MHz network: the helicopter, the remote control and the base station on the ground. To ensure the deterministic delivery of the data packets, transmit collisions must be avoided. This is achieved by having the devices synchronize with the helicopter using the packets sent by it as a reference. During the 10 ms sampling period of the control system, each of the three devices present on the network has its own window assigned when it sends data. They keep their own internal timing so they remain synchronized and are able to send data at the right time even if some radio packets transmitted by the helicopter are not received due to radio interference.

The 802.11n network connection is made using an Intel 4965AGN mini PCI express wireless card installed on the RTM PCB (Fig. 8, 9) but controlled by the computer module running Red Hat Enterprise Linux. It offers support for three antennas. Currently only one vertical antenna is installed on the helicopter body, but in the future two more will be installed orthogonal to the existing one to allow reliable data transfer during aerobatic maneuvers when the orientation of the helicopter changes.

2.8. Vision System

Video cameras, especially when installed in a configuration for stereo vision, can be cheap sensors that provide a great amount of information useful for control and navigation. Currently our interest is focused on system identification and nonlinear control of the helicopter, where video cameras are not absolutely required. However the electronic system was designed with cameras in mind. Prosilica GC650C cameras were considered for this application. They communicate using the GigE Vision Standard on Gigabit Ethernet. The video data is provided directly in digital format and uncompressed. The processor overhead for capturing the data using the network adapter is about 3% when jumbo frames are used. This is much better than using USB where due to the small data packets a large amount of CPU time is taken just to transfer data to memory. Currently the ETXexpress computer module has only one Gigabit Ethernet connection. A small external Ethernet switch is necessary to connect both cameras simultaneously. In the future designs, a dual-port PCI express Gigabit Ethernet controller will be implemented on the RTM PCB to avoid using the switch, or an extra DSP board with two Ethernet connections will be added.

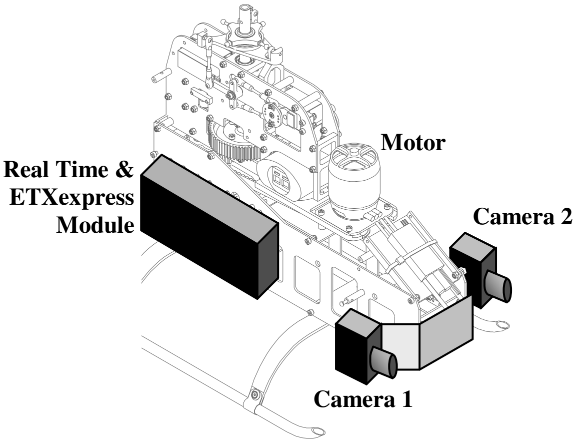


Figure 15. Placement of the video cameras on the helicopter body

Stereo vision requires very precise synchronization of the trigger of the two cameras. Most if not all of the USB cameras don’t provide such a feature. The GC650C cameras allow for an external trigger input and can also generate trigger signals. This way, one camera can generate the trigger and the second can use it for synchronization, with no additional logic needed. Other parameters such as the exposure time, white balance and gain can be configured simultaneously over the Gigabit Ethernet interface by the software running on the computer module. Another important feature needed for fast-moving vehicles is to have progressive (non-interlaced) image capture, otherwise the images would be almost unusable.

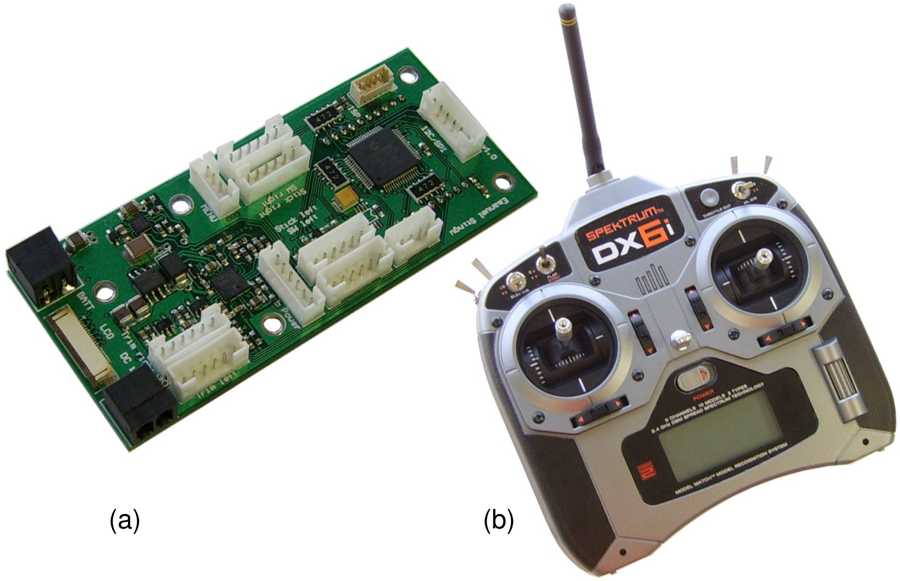
A third small USB camera will be used for landing on a moving platform installed on a ground vehicle. The camera will point downwards to get the pattern drawn on the platform, which will be used by the vision processing algorithms to extract the orientation and the distance of the helicopter relative to the ground vehicle.

To maintain the center of gravity high enough so aerobatic maneuvers are not affected, the cameras are not installed under the helicopter body using elongated skids, but on the sides (Fig. 15). This approach limits the movements the cameras can do without being obstructed by the helicopter body, but it can be used especially if the cameras are fixed.

2.9. Remote Control Unit

The experiments involving the helicopter need the intervention of a human pilot. It could be just the need to do simple testing of the basic functionality of the helicopter platform, to activate the automatic control functions or to take over in case of various failures and land safely. The use of remote controls from the RC helicopter hobby industry adds an unnecessary risk factor into the system. It is well-known that many of them are sensitive to radio interference. Some newer models that use the 2.4GHz frequency range solve this problem, but have limited range.

A completely new electronic board was designed for the remote control (Fig. 16). A Spektrum DX6i model was used as the platform that had to be customized. It contains the enclosure, buttons, joysticks and the graphical LCD screen. The original electronic board was replaced with the new design. The low-capacity Ni-MH batteries that were provided can not be charged unless they are completely discharged, limiting the usage flexibility. They were replaced by a higher capacity Li-Po battery that allows the remote control to work for several experiments without the need to be recharged. The electronic board implements a Li-Po battery charger, so the battery doesn't have to be removed from the enclosure each time it is charged.



(a)

(b)

Figure 16. Remote control. New electronic board (a) and the Spektrum DX6i (b) – not to scale

The potentiometers of the joysticks, the buttons and the LCD connect to the new board. An XTend RF transceiver operating on 900 MHz allows commands to be sent to the helicopter and important parameters to be displayed to the pilot on the LCD screen. The voltage and the current of the battery are continuously sampled to estimate the remaining operating time and to give warning to the pilot before the battery runs out.

2.10. Base Station

The Base Station (Fig. 17) is an optional component of the control system. The on-board computer of the helicopter can run the control algorithms by itself and can store the history of all the signals in the system. The main function of the base station is to display detailed information to the operators on the ground, to give high-level commands to the helicopter and to provide reference ground pressure and DGPS corrections.

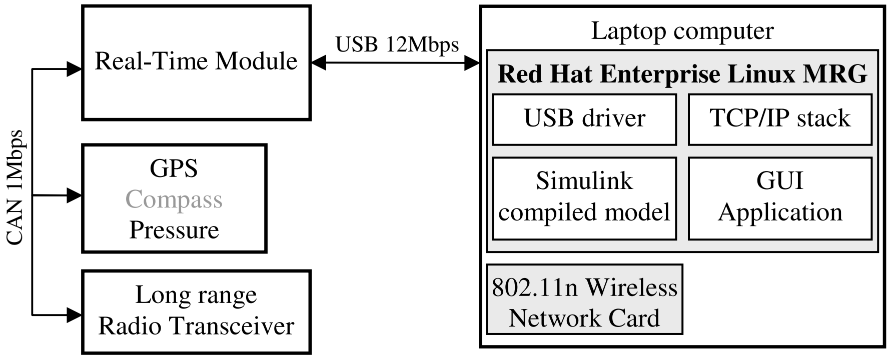


Figure 17. Organization of the ground monitoring station

The electronic system of the helicopter was designed to be modular and reusable. The base station does not need additional hardware. It can use a set of the modules already available for the helicopter. There is no need to use an ETXexpress computer module on the ground. A laptop computer is more functional. The RTM can interface via the USB bus with the laptop computer the same way it does with the ETXexpress module.

CHAPTER 3

HETEROGENOUS MULTI-VEHICLE CONTROL

The helicopter platform will be included in an “ecosystem” of different types of vehicles. Currently in the development phase there are a relatively large size ground vehicle and a quad-rotor helicopter (Fig. 18).

The ground vehicle is built using a 4x4 all-terrain wheelchair base (the X4 Extreme model, manufactured by Innovation in Motion). It has four independent motors that drive each wheel separately and a passive steering system for the front wheels. The driving range is 19 miles and the maximum speed is 5.5 mph. It will be used as a dynamic landing platform for the helicopters.

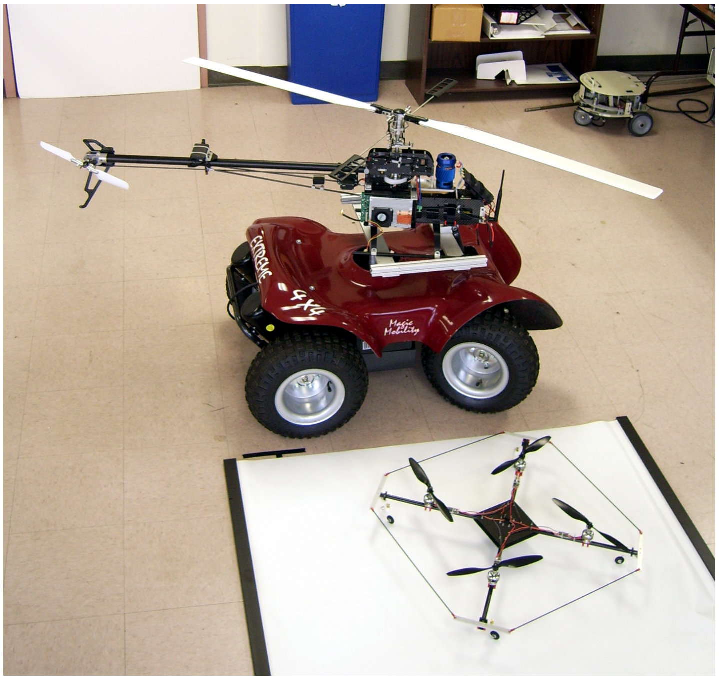


Figure 18. Three types of vehicles for multi-vehicle control

In Fig. 19 it can be seen that most of the electronic components are those already designed for the helicopter. The ground vehicle can carry a heavy load, so there is no reason to optimize the system for low weight by using an ETXexpress computer module. Instead, a laptop computer is installed as in the case of the base station. The long-range radio transceiver, the inertial unit and the GPS-compass sensor module are still necessary for the control algorithm. The movement of the wheels and the steering direction will be measured using custom-made incremental magnetic encoders. They are much more robust to vibrations, mechanical displacements and dust compared to the optical encoders and can achieve the same precision for this application. The motors are controlled by a high-power speed controller that is installed by default on the platform. The original control system used the CAN bus to communicate with the speed controller, so no changes are necessary to interface it to the RTM.

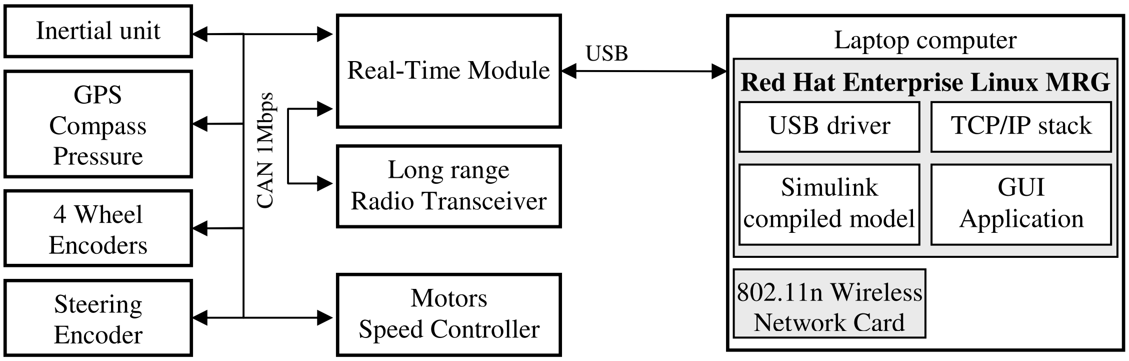


Figure 19. The electronic system on the ground vehicle

The design of the electronic system for the quad-rotor helicopter is different because it does not have the possibility to carry a heavy payload. Therefore a powerful on-board computer can not be implemented. The control algorithm is executed on a ground base station. The on-board electronic system measures different system states, sends them to ground using a radio transceiver and, after the control algorithm processes the information, receives the commands that have to be given to the motors. The block diagram can be seen in Fig. 20.

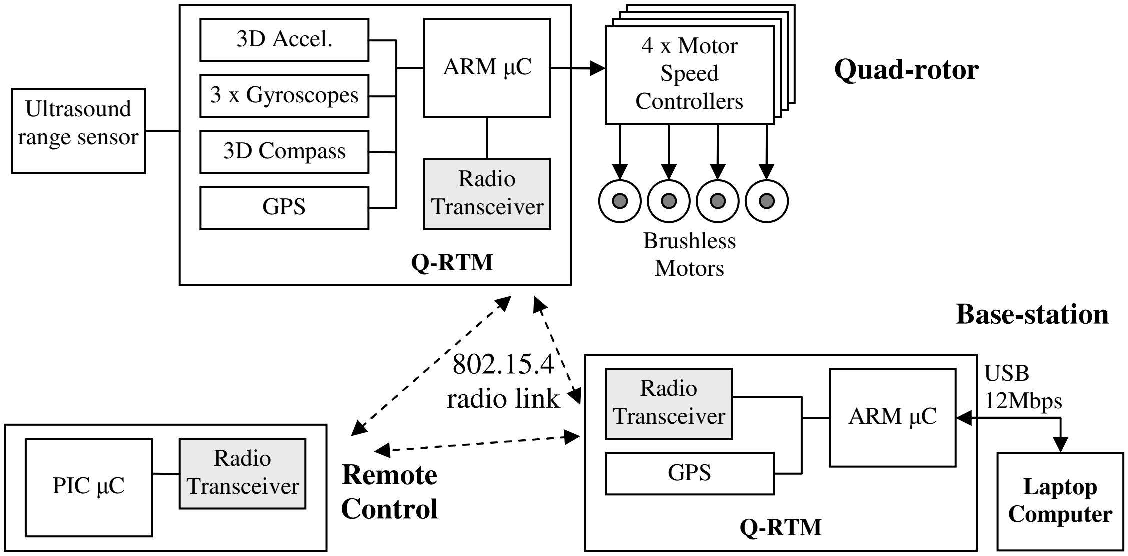


Figure 20. The electronic system for the quad-rotor helicopter

The design of the helicopter electronic system was done using a highly modularized hierarchical electronic schematic. It was very easy to select the parts needed from different modules and put them together on a single PCB (Fig. 21) in order to create the Quad-rotor Real-Time Module (Q-RTM). The inertial sensors are less precise than in the case of the helicopter, but much easier to interface.

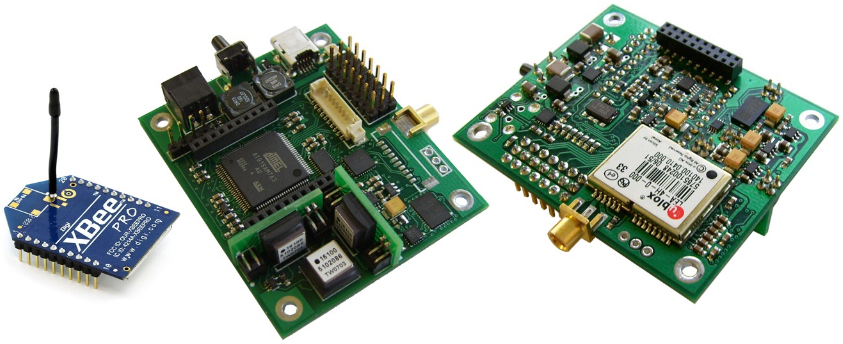


Figure 21. The quad-rotor RTM

The radio transceiver is an XBee-PRO module manufactured by Digi International. It operates on 2.4GHz and has a RF data rate of 250 Kbps. The line-of-sight range is about 1.6

miles. This model was preferred to the XTend RF module used on the helicopter due to the higher data rate and the smaller size, the range being of secondary importance. The hardware and software interfacing is practically the same. The ARM processor is again the same as on the helicopter RTM. This allows most of the software to be shared on the two platforms. The remote control uses an identical electronic board as the helicopter, but instead of the XTend module it has an XBee-PRO module installed. This module is designed to communicate using the ZigBee protocol, but in this mode it can not achieve real-time performance. Instead, the lower-level 802.15.4 protocol (on which ZigBee is based) is used. To avoid latencies and possibly long dropouts in communication, the transmission mode is set not to wait for any acknowledge from the destination. The three devices present in the network use the same synchronization mechanism as in the case of the helicopter long-range transceiver to avoid collisions while transmitting.

The multi-vehicle control is realized using the 802.11n wireless network to transmit the necessary data between any two entities through UDP packets. Each helicopter, ground vehicle, their base-stations and the quad-rotor base-stations behave the same relative to this network, allowing for a unified framework where the vehicle type is not important.

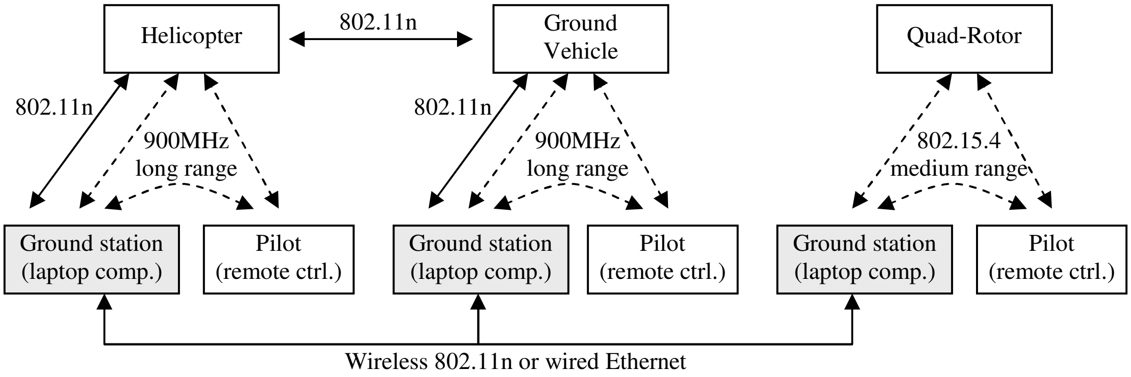


Figure 22. The radio communication between different vehicles

CHAPTER 4

REAL-TIME CONTROL

Hard real-time systems consider the late delivery of correct results as a system failure. The control system of the UAV helicopter is such an example, where both the correctness of the results and their timely delivery must occur for the aircraft to operate safely. The hardware and the software must guarantee the deterministic execution of the control loop. The main operations that have to take place during one sample period are shown in Fig. 23. For slow systems where the sample frequency is low, the entire control process can be implemented on an easy to use operating system, like Microsoft Windows or Linux. For high sample frequencies, a real-time operating system is mandatory for deterministic execution. Usual choices are QNX Neutrino or VxWorks, the latter being used on the Spirit and Opportunity Mars Exploration Rovers. They provide very good support for real-time applications and enhanced reliability compared to the mainstream operating systems. A shortcoming is the lack of support for hardware. Although they come with drivers for an entire range of peripherals, it is difficult to extend the functionality to devices that are not already supported.

Because of the ease of use and the broad support for hardware, real-time capabilities were added to Windows and Linux. The main approach was to add a second real-time kernel that can preempt the main kernel (well-known examples are RTLinux and RTAI). This allows Windows or Linux to run as a task in the new real-time kernel. The determinism is very good, but the initial problem related to the hardware support is only partially solved. A real-time task is not allowed to make Windows or Linux system calls. In order to make use of the hardware support, like network devices, USB peripherals and storage devices, the real-time task has to send the request to a Windows or Linux task that in turn forwards it to the appropriate driver. Anything running in the Windows or Linux context is not deterministic. A control loop can be implemented in the real-time kernel, but it still can not make use of the Windows or Linux services in real-time. This might be fine with some applications that don't require deterministic performance for the network or USB communications, but is a major limiting factor for the current project.

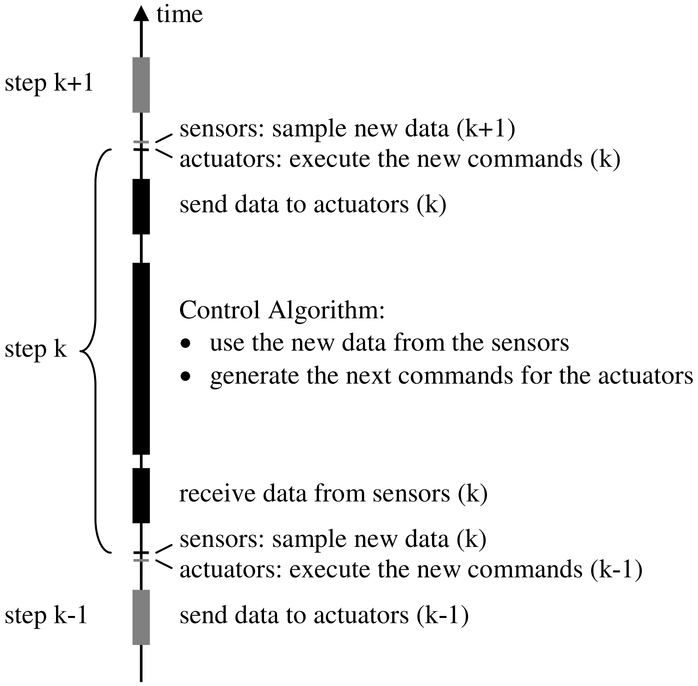

time
step k+1
sensors: sample new data (k+1)
actuators: execute the new commands (k)
send data to actuators (k)
Control Algorithm:
• use the new data from the sensors
• generate the next commands for the actuators
step k
receive data from sensors (k)
sensors: sample new data (k)
actuators: execute the new commands (k-1)
step k-1
send data to actuators (k-1)

Figure 23. Temporal succession of events in a sampled system

In the last few years there were efforts to make the 2.6 Linux kernel real-time [12]. Ingo Molnar (now an employee of Red Hat) provides a patch that improves the responsiveness of the kernel dramatically. In December 2007 Red Hat has released a beta version of Red Hat Enterprise MRG (Messaging Real-time Grid) that includes the optimized kernel. The maximum latencies have been improved two orders of magnitude compared to the standard Linux kernel to the level of a few tens of microseconds.

Each computer (on the vehicles and on the base stations) runs Red Hat Enterprise Linux MRG. The control is provided by a multi-threaded process that handles USB communications with the Real-Time module, network communications with the other base station(s) and/or the other vehicles and executes a compiled Simulink model with the control algorithm. On the base stations, a separate process with a lower priority is used to display the necessary information to the pilot and to accept high level commands. The non-compiled Simulink model can also run on the base stations, in the Matlab environment. It can receive the real-time data and can change various parameters of the model using the wireless network.

The control algorithms are implemented in Simulink (Fig. 24). Special blocks were created for the sensors and actuators. There are also blocks that represent an entire helicopter, a ground vehicle, a base station or a pilot remote control, such that formation control algorithms can be implemented very easily. The Simulink model is converted to C code using the Real-Time Workshop. A new target is created specifically for this system. The code is included and compiled into the real-time Linux application, which runs on the ETXexpress computer module or on the laptop computer. The application provides the Simulink blocks with all the necessary data obtained from the Real-Time Controller or from the wireless network. Each block has a set of configuration parameters that are processed by the real-time application when the Simulink model is initialized. The sensors and the vehicles have unique identifiers which make it easy to associate a specific block in Simulink to a specific sensor, actuator or vehicle.

The RTM uses USB to send a data record containing the readings of all the sensors to the computer (Fig. 25). This is done periodically at the sample rate. The control software passes the data to the compiled Simulink model, executes one iteration of the algorithm and the result is sent back to the RTM using USB before the end of the current sample period to give commands to the actuators.

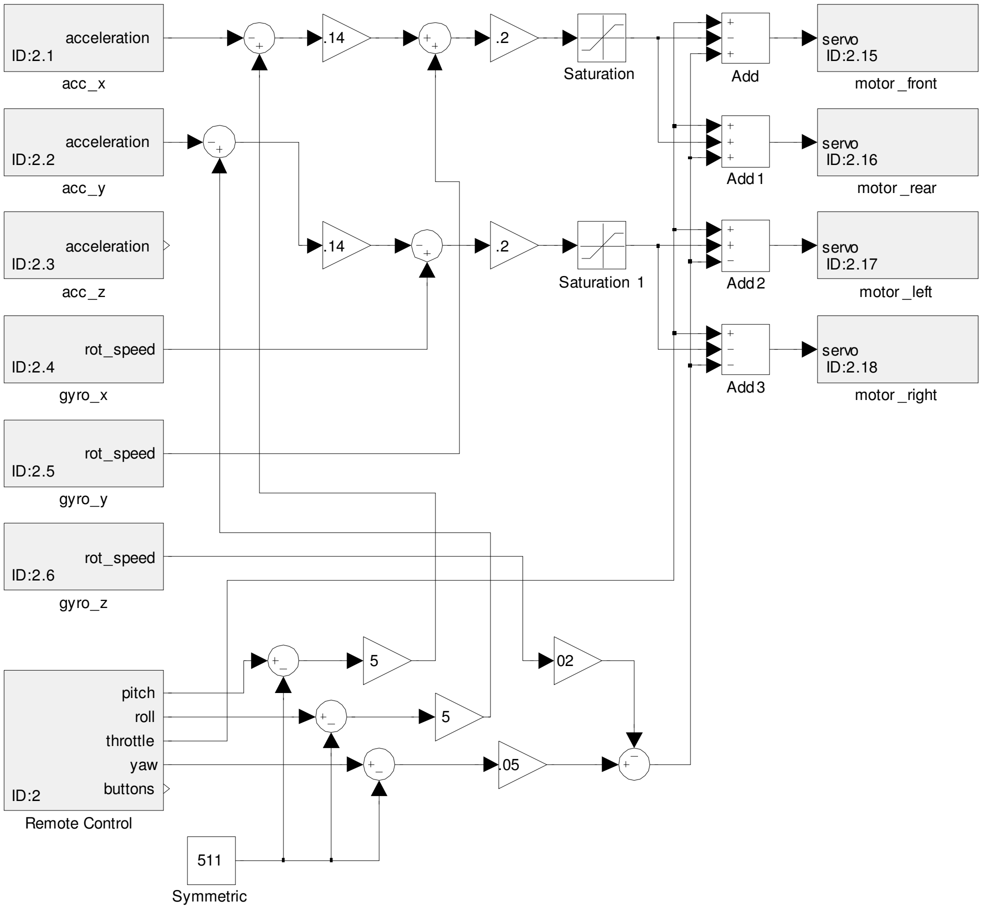


Figure 24. A simple implementation of a controller for the quad-rotor in Simulink

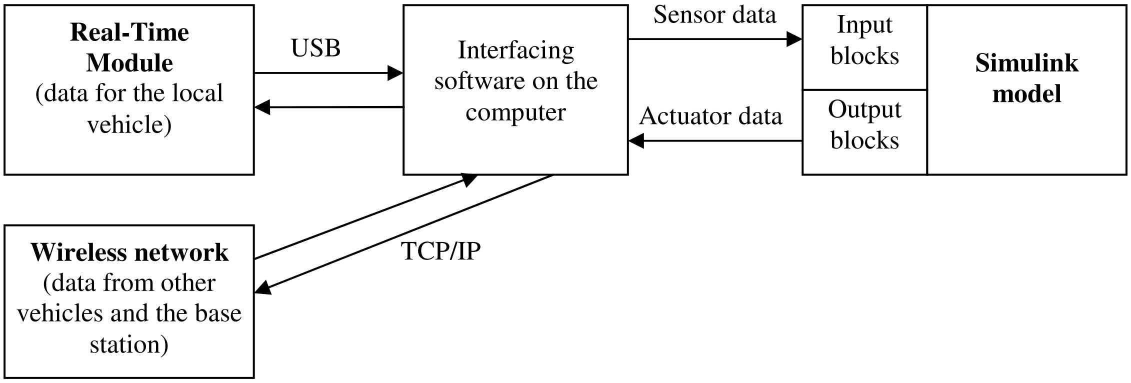

Real-Time Module
(data for the local vehicle)
USB
Interfacing software on the computer
Sensor data
Actuator data
Input blocks
Output blocks
Simulink model
Wireless network
(data from other vehicles and the base station)
TCP/IP

Figure 25. Data flow on the computer

CHAPTER 5

CONCLUSION AND FUTURE WORK

Research topics in UAV control become more and more complex. The industry and the Army are reluctant to accept complicated theoretical solutions without a practical implementation to prove their applicability. This thesis presented a hardware platform that can be used to implement aerobatic UAV control, formation flight and aerial-ground vehicle coordination in order to help validating the theoretical results in practice. Because most of the electronic system and the associated software are custom-developed by the author, the system can be adapted to new requirements without any restrictions that may otherwise appear due to proprietary hardware or software. Further, because many building blocks, both electronic and software-related, are already available, new modules can be designed considerably faster.

The future work will combine experiments using the various vehicles with improvements to the software environment. The goal is to have a system that is easy to use by any person, without detailed knowledge of the hardware implementation.

The first experiment will be to identify the linear model of the helicopter for hover. The helicopter stand will be used to prevent any crash. Having the linear model is very important for further development of any type of controller. Based on this model, a simple controller that can enhance the stability of the helicopter will be designed. It will be used as a backup controller later when experimenting with more complex solutions. It will be implemented on the Real-Time Module because it will have to work even in the situation when the on-board computer fails.

Electric helicopters are good platforms for testing different system identification methods. A good nonlinear model of the helicopter is required for developing aerobatic flight controllers or for simulations. Because they do not use fuel, the mass and the inertia moments are constant during flight and during experiments, allowing a consistent behavior every time.

Combustion engine helicopters are much more difficult to model precisely due to the fact that the mass and the inertia moments change during flight. The fuel consumption has to be modeled, but it is difficult without having the necessary sensors for the fuel-related parameters. Once experience is gained with identification of electric helicopters, fueled helicopters will also be considered.

Once the nonlinear model of the helicopter is obtained, a hardware in the loop framework will be developed. Complex maneuvers need a good visualization platform. A flight simulator will be interfaced with the Matlab and Simulink environment for a better representation of the simulations and for replaying the recorded flight data.

The sensors available on the helicopter body are sufficient for the stabilization of the platform, but do not provide the necessary information for more complex behaviors. Automatic landing and take-off, landing on a moving vehicle, obstacle avoidance, SLAM and many other applications require measurements of different characteristics of the environment. The video cameras on the helicopter body can provide a lot of information, but this requires the development and implementation of video processing algorithms. This will be a research topic in the future.

When controllers with good performance are implemented for the different vehicle platforms, research will continue with the development and implementation of formation control algorithms and vehicle synchronization. Because they are easy to model and to control, the quad-rotors and the ground vehicle will be mainly used for these algorithms.

REFERENCES

[1] V. Gavrilets, B. Mettler, E. Feron, “Nonlinear Model for a Small-Size Aerobatic Helicopter”, Proc. of the AIAA Guidance, Navigation and Control Conference, Montreal, Quebec, Canada, 2001.

[2] C. L. Castillo, W. Alvis, M. Castillo-Effen, W. Moreno, K. Valavanis, “Small Unmanned Helicopter Simplified and Decentralized Optimization-based Controller Design for Non-aggressive Flights”, International Journal on Systems Science and Applications, 2006.

[3] G. Vachtsevanos, L. Tang, G. Drozeski, and L. Gutierrez, “From Mission Planning to Flight Control of Unmanned Aerial Vehicles: Strategies and Implementation Tools”, Annual Reviews in Control, 2004.

[4] Cai, G. Chen, B.M. Peng, K. Dong, M. Lee, T.H., “Modeling and Control System Design for a UAV Helicopter”, MED ’06, 14th Mediterranean Conference on Control and Automation, June 2006.

[5] V. Gavrilets, B. Mettler, E. Feron, “Dynamic Model for a miniature aerobatic Helicopter”, MIT-LIDS report, # LIDS-P-2580, 2003.

[6] O. Amidi, T. Kanade, R. Miller, “Vision-Based Autonomous Helicopter Research at Carnegie Mellon Robotics Institute”, Proceedings of Heli Japan ‘98, Gifu, Japan, April 1998.

[7] E.N. Johnson, D.P. Schrage, “System Integration and Operation of a Research Unmanned Aerial Vehicle”, AIAA Journal of Aerospace Computing, Information, and Communication, Vol. 1, No. 1, January 2004.

[8] E.N. Johnson, D.P. Schrage, J.V.R. Prasad, G.J. Vachtsevanos, “UAV Flight Test Programs at Georgia Tech,” Proceedings of the AIAA “Unmanned Unlimited” Technical Conference, Workshop, and Exhibit, 2004.

[9] M. Dong, B. M. Chen, G. Cai and K. Peng, “Development of a real-time onboard and ground station software system for a UAV helicopter”, AIAA Journal of Aerospace Computing, Information, and Communication, Vol. 4, No. 8, pp. 933-955, 2007.

[10] D. H. Shim, H. J. Kim, S. Sastry, "A Flight Control System for Aerial Robots: Algorithms and Experiments," IFAC Control Engineering Practice, 2003.

[11] CAN in Automation Group, “CANopen. Application Layer and Communication Profile”, Draft Standard 301 Revision 4.2, 2002.

[12] D. Hart, J. Stultz, T. Ts’o, “Real-time Linux in real time, IBM Systems Journal”, Vol. 47, No. 2, 2008.

[13] E. Stingu, F. L. Lewis, “A Hardware Platform for Research in Helicopter UAV Control”, Proc. International Symposium on Unmanned Aerial Vehicles, Paper 18, Session II, UAV 08, Orlando, June 2008.

BIOGRAPHICAL INFORMATION

Emanuel Stingu received his Bachelor of Engineering degree in Automatic Control from the Faculty of Automatic Control and Computer Engineering, Technical University of Iasi, Romania, in 2006. In the same year he started his Masters in Electrical Engineering at the University of Texas at Arlington under the supervision of Dr. Frank Lewis. He has been working as a Graduate Research Assistant at the Automation & Robotics Research Institute.

He showed interest in electronics and computer programming ever since he was in the 5th grade in elementary school, winning top places in Electronics and Physics competitions at the national level in Romania. His experience allows him to easily combine the control theory with the requirements of practical systems. His interests are mostly in nonlinear control, including modeling, system identification and robust/adaptive control.