World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:7, No:6, 2013

Integrate Communication Modeling into the Design Modeling at Early Stages of the Design Flow Case Study: Unmanned Aerial Vehicle (UAV)

Ibrahim A. Aref and Tarek A. El-Mihoub

Abstract—This paper shows how we can integrate communication modeling into the design modeling at early stages of the design flow. We consider effect of incorporating noise such as impulsive noise on system stability. We show that with change of the system model and investigate the system performance under the different communication effects. We modeled a unmanned aerial vehicle (UAV) as a demonstration using SystemC methodology. Moreover the system is modeled by joining the capabilities of UML and SystemC to operate at system level.

Keywords— Modelling, SoC, SystemC, UAV, Simulation, SoC.

I. Introduction

IN recent years, the increasing technological complexity of wireless communication systems coupled with requests for more performance and shorter time to market have produced a challenge to develop System-on-a-Chip (SoC) design methodology to support those complex systems. Under such conditions, it has become essential to used SoC design methodologies such as SystemC methodology [10], [11] to model and design such complex systems. A system on chip (SoC) [6], [8] is a complex integrated circuit that integrates the major functional elements of a large system into a single chip. It may integrate different types of components such as digital elements, electronic elements or non electrical parts (sensors) as well as software components. The design and the verification of such systems require powerful modeling and simulation techniques to address all aspects consistently and efficiently. In particular systems such as complex wireless communication systems, designers are facing increased difficulties in managing the large complexity inherent in those systems. SystemC is emerging as a suitable design and modeling language, because it provides a consistent methodology for the design and refinement of complex digital systems. This methodology is essential to manage complexity and enhance designer productivity. It allows the designer to view designs at different levels of abstraction, and in particular advocates the evaluation of the system performance early in the design cycle, and its use in guiding the refinement process into lower levels of abstraction.

In this paper, we need to show how we can integrate communication modeling into the design modeling at early stages of the design flow. We consider effect of incorporating noise such as impulsive noise on system stability. Furthermore, we show that with change of the system model and investigate the system performance under the different communication effects. Therefore, to study such conditions, we modeled an unmanned aerial vehicle (UAV) [7], [12] as a demonstration using the wireless channel model [1]. As mentioned above, our target here is not just design and implements a UAV, because it is just an application, the implementation of such system is look like a trivial and there are many people that design and implemented that system before. But in this work, the main goal is how we can optimize the system stability in terms of communication. In the beginning, we constructed the system at high abstraction level of communication, i.e. the communication between Ground Control Station (GCS) and Unmanned Aerial Vehicle (UAV) is done using variables. At this level, we used a modeling language based on UML (Unified Modeling Language) [9], [3] notations for high level system descriptions. The system has been modeled successful and we got a good result that represents system behavior. Moreover, we incorporate wireless communication channel (without and with the noise) in order to investigate stability and reliability of the system to get the best performance under the communication effects.

Unified Modeling Langue (UML) [9] is widely accepted as a standard notational language for visualizing, specifying, constructing and documenting software systems. It is a common framework for communicating and capturing different architectural views. Through providing a variety of diagrams, it helps to visualize the software systems from different perspectives [3]. It underpins object-oriented design and promotes reusability. Furthermore, UML profiles provide extensions to the language to develop new diagrams. It also defines a set of extensions to meet the demands of specific application domains. In spite of being originated to serve the software engineering community, it is also used to describe real time systems. Many enhancements to the UML standards

I. Aref and T. El-Mihoub are with Computer Engineering Department., University of Tripoli, Tripoli – Libya (e-mail: iaref@tripoliuniv.edu.ly & t.mihoub@tripoliuniv.edu.ly).

aim to facilitate modeling complex real time embedded applications.

On the other hand, SystemC [13] can be used to abstract applications and platforms at sufficiently high levels in addition to enabling the linkage to hardware implementation and verification. SystemC, as an object-oriented programming language, has a strong correlation with UML. S. Despite being at present a hardware description language, the enhanced version in the making of SystemC will support software module descriptions and run time features including scheduling. SystemC has the potential to fully-fledge describe execution platforms that can serve as the target of a codesign methodology. The SystemC is expected to success in being a preferred intermediate description language.

SystemC [5] cannot be as efficient and effective as UML in abstracting applications at high levels. In the process of requirement engineering, visual notations facilitate the interacting with the end-users and ease the requirement elicitation and analysis. It is also important to be able to use standard models of computation (MOCs) at the early design stages. Furthermore, in this stage, specifying communication mechanism is not crucial and its definition can be left for the underlying operational semantics of the MOCs being deployed.

The rest of the paper is organized as follows: Section II gives a brief description about modeling complex wireless system based on SystemC methodology. Section III presents a demonstration of a unmanned aerial vehicle (UAV), Section IV presents simulation results and discussion, and finally Section V presents conclusions.

II. Modeling Complex Wireless System Based on SystemC Methodology

Complex wireless communication systems set various challenges in modeling and design fields. Inserting communication as a new dimension in the design flow process causes conventional SystemC methodology tasks to become more complex, and becomes SystemC wireless methodology [4], which contains system and communication aspects. Modeling of communications and their interaction with system aspects is thus key for effective design methodology in the early stages of design flow. In this work, UAV is selected as a demonstration. This is a very complex system modeled based on the developed methodology and partitioned along different parameters. By applying SystemC wireless methodology [4] to model this system, we can prove that incorporating and fixing the wireless channel, wireless protocol, noise or all of these elements early in the design methodology is highly advantageous. The main point is the integration between computation, communication and SystemC methodology, i.e., introducing integrated communication modeling into the design modeling early in the system development. This allows us to investigate the system very easily and make changes quickly, because SystemC is a unified environment, which means everything is on the same platform. The modeled system is introduced to simulate the behavior of whole system. Wireless communication between Ground Control Station (GCS) with Unmanned Aerial Vehicle (UAV) is addressed using a wireless channel model to establish communication, thus in this case wireless communication between GCS and UAV is addressed using a Point-to-Point (P2P) channel [2].

III. Case Study: Unmanned Aerial Vehicle (UAV)

In this section, a demonstration is created in order to validate the developed SystemC methodology. We modeled a UAV as case study using the wireless channel model [1]. We present a system level description based on UML-notations from which one can extract SystemC code. Our modeling framework is based on a restricted set of UML diagram types, which means the system specifications of the modeled system are developed using the UML-compatible tool. We then refine the designed model that implemented by SystemC code. Consequently, we can describe executable platforms of the system at the UML-level as well as translate UML-based application descriptions to SystemC level. We show in Fig. 1 how to model a system in UML and generate the system executable model in SystemC.

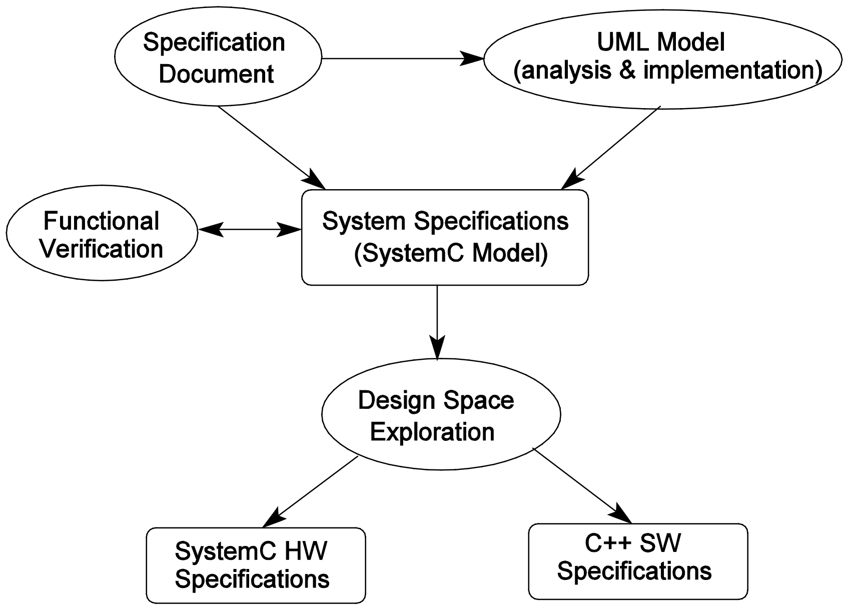
Specification Document
UML Model (analysis & implementation)
Functional Verification
System Specifications (SystemC Model)
Design Space Exploration
SystemC HW Specifications
C++ SW Specifications

Fig. 1 UML SystemC

A. UAV Subsystems Overview

Unmanned Aerial Vehicles (UAVs) [7], [12] are crafts capable of flight without an onboard pilot. They can be controlled remotely by an operator, or can be controlled autonomously via preprogrammed flight paths. They are used by the military for recognizance and search and rescue operations. In this work, we present a SoC design methodology joining the capabilities of UML and SystemC as illustrated in Fig. 1 to operate at system level. Also we need to show how we can integrate communication modeling into the design modeling at early stages of the design flow. UAV is chosen as case study to achieve that target.

As shown in Fig. 2, UAV system contains a Ground Control Station (GCS) with Unmanned Aerial Vehicle (UAV). GCS is used to steer UAV. The commands are delivered from GCS to UAV. The vehicle should be reaching a desired target through the control from GCS. UAV has to go from a starting point (point of departure - (x1,y1,z1)) to ending point (final destination-(x2,y2,z2)). At the final destination, UAV have to get some pictures. Finally, UAV have to return to the point of departure.

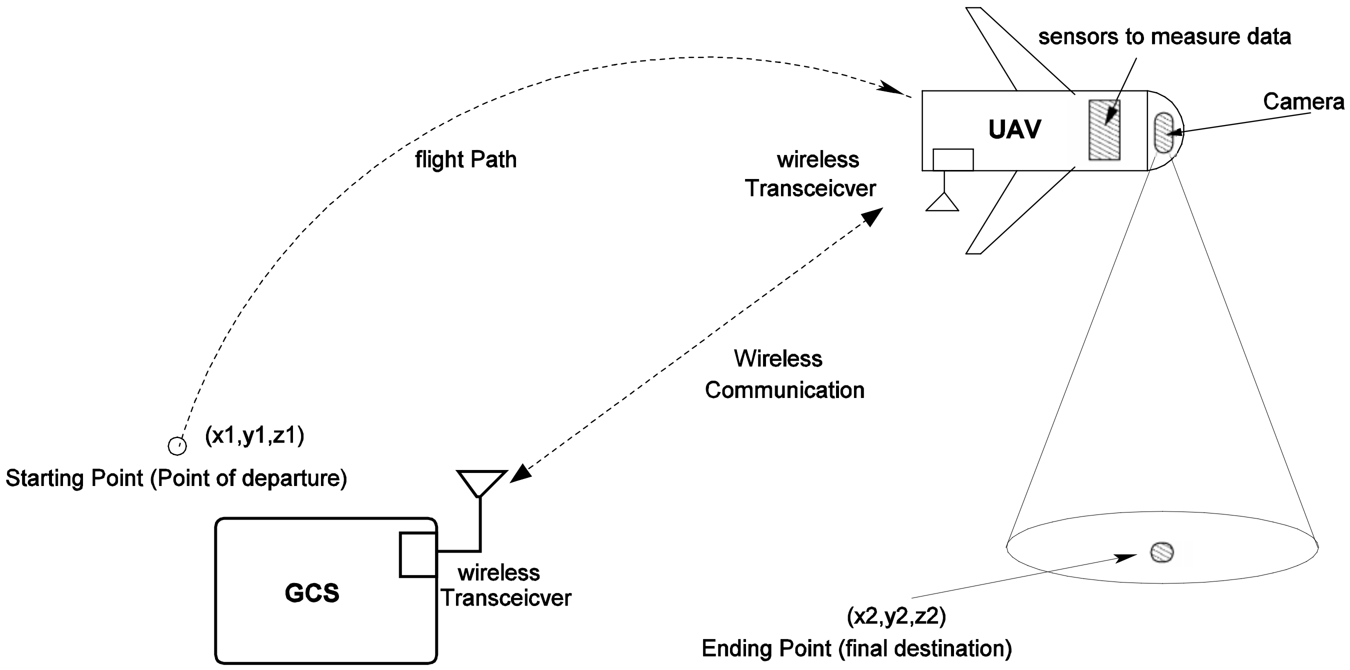

Fig. 2 Block Diagram of the UAV System

The UAV system can be operated in two modes, the first one is manual control mode, which is the UAV vehicle is fully controlled by a human pilot, and the system serves only as a measurement device. The other mode is automatic control mode, which is UAV vehicle takes over some or all actuators. The communications between UAV and GCS are provided by data link. For real-time sensing the data link transfers control signals from the GCS to the UAV and sensed data with vehicle condition parameters from the UAV to the GCS. The main functions of the data link are: control of the UAV, status of the UAV, control of the payload, status of the payload and sensed information from payload.

B. UAV System Specification Model

A specification model, which is known sometimes as executable specification, is the first phase to design a system with complete functionality (pure functionality). It is a pure behavior description. In this model, the functionality of the system has been modeled without introducing any unnecessary implementation details. No timing characteristic is modeled for the computation and communication behaviors. When simulated, the specification model is assumed to be executed in zero time. The specification model of this work was derived from the C++ description that created without introducing any implementation details. The system specifications are achieved based on the following scenario:

- UAV has gone from a starting point (a1) to the target (a2) as shown in Fig. 3.
- In the target location, UAV have to get some pictures under control commands from GCS, which means, in this work we focus only on manual control mode.
- Finally, UAV have to come back to the original point.

To do this, we need in the beginning to write an abstract specification using a high level language, because simulation of this executable functional specification allows us to check the system functionality. Moreover, writing a C++ description can help us to understand system behavior and to decide system requirements and constraints exactly.

1. UAV System Description

The behavior of the system that we are going to model is as following:

- GCS generates the starting point and ending point locations with initial speed that needed for takeoff operation.
- GCS calculate distance and find the suitable path to the target based on given locations and then send data to UAV.
- GCS decide the time period that UAV have to send sensors measured data (current location, speed and altitude) to GCS.
- When UAV reach the destination point (target), it have to complete the specific task that it sent for it (for example getting some pictures).
- Finally UAV have to come back to the original point (departure point).

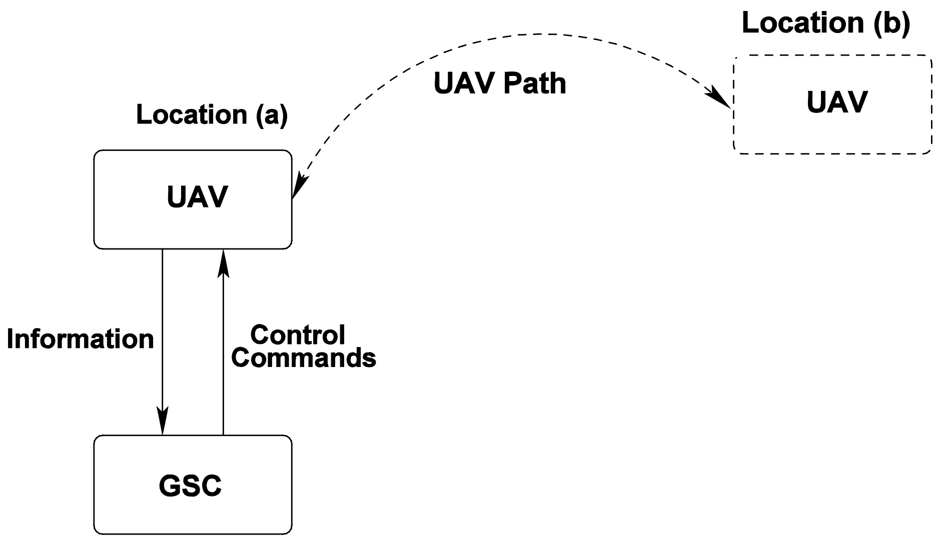

Fig. 3 UAV System Specifications

The UAV system use-case diagram of functional requirements (for manual control mode) can be seen in Fig. 4. A prototype of a UAV system with limited functionality is created. This prototype is implemented in C++. It had the aim to further increase the knowledge of the UAV system as well as the main aim which is create functional specifications with SystemC. The communication between the functional blocks (objects) are implemented by using global variables rather than channels since up to now no concurrency and synchronization characteristic is specified.

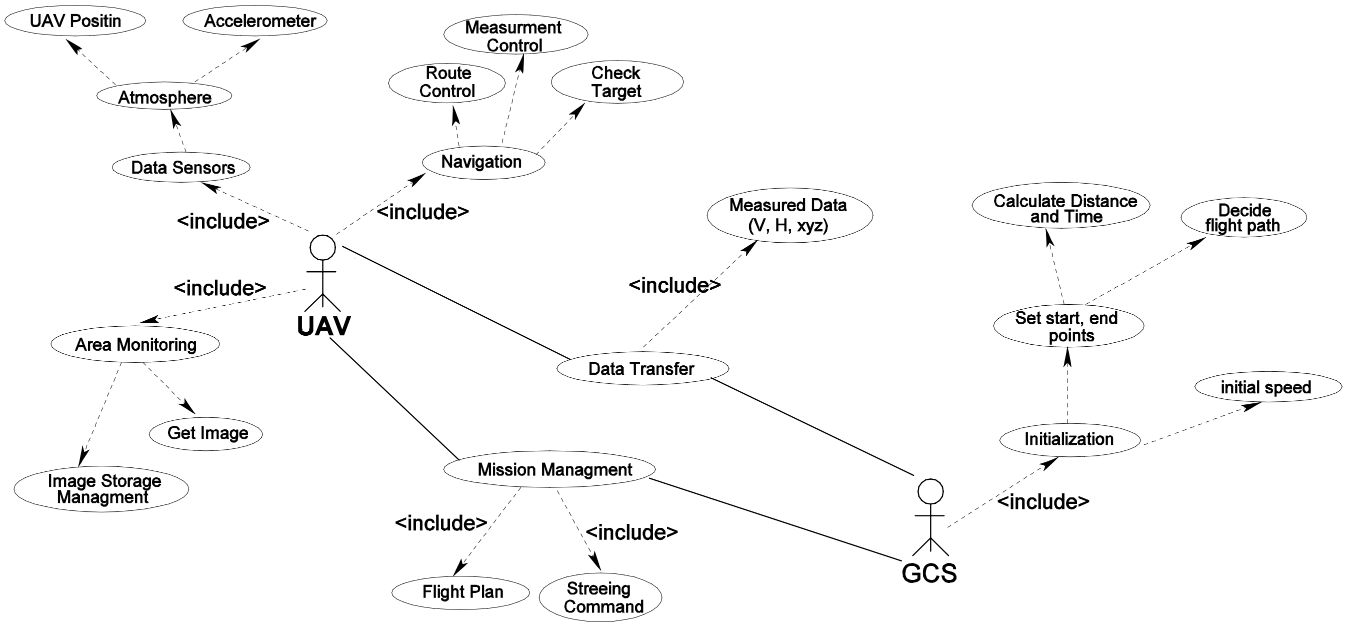

Fig. 4 UAV System use-case diagram of Functional Requirements

2. UAV System Functionality

All UAVs have some common core of functionality. This functionality can be illustrated as shown in Fig. 5 with different level of abstraction (from high level to low level) as following:

- Level 1: Decide a target, then go there and get some pictures.
- Level 2: Decide a target, then a navigation process which is following a route through space. After that fly which is move the control surfaces (UAV) in response to navigation commands (sensors). Finally, at the target point, take some pictures.
- Level 3: Fly which is moving the control surfaces (UAV) in response to navigation commands (sensors). Takeoff and Landing. Navigation which is follows a route through space. At the target point, take some pictures. Autopilot which is move or proceed the control surfaces in response to sensors. Uplink which is turn buttons in GCS into actions on the UAV. Downlink which is turn numbers in UAV into displays in GCS. Finally, turn things ON or OFF in response to commands or automatically.

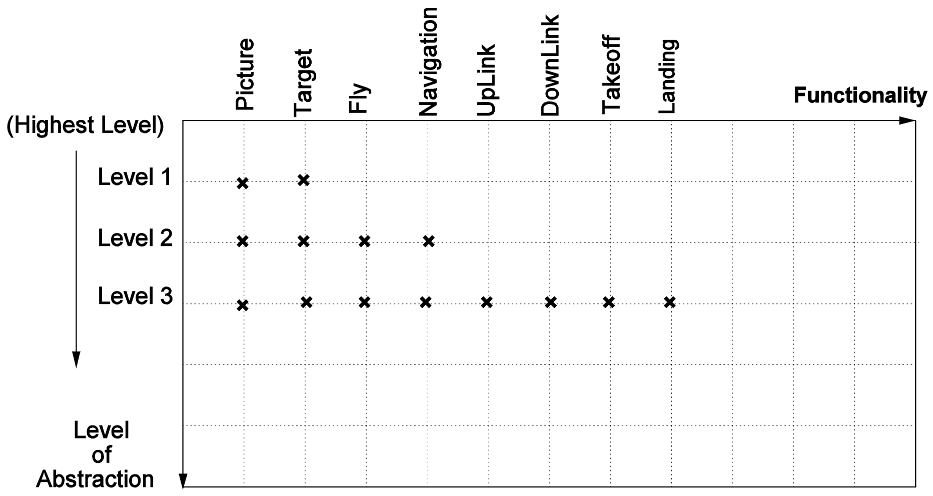

Fig. 5 Functionality Refinement Levels

C. Architecture Model

It is an refined model from the specification model with the partition of hardware and software (architecture exploration). The refinement process is done by mapping functions to specific hardware blocks (architectural/physical blocks which are pre-existing IP or new IP). In this work, we have created this model by divided the functionality into many independent processes, which allows for concurrent functions. We have developed more than one options and than we can make a comparison based on performance metrics that will discuss later. We can select the best architecture based on comparison process. So to get this model, we have to make the architecture exploration which contains the following steps:

1. Allocation and Functions Partitioning

In this phase, we have to map each function into component. The UAV system is divided into three blocks based on the functionality observed in the C++ code. In the top level, there are three blocks (modules), GCS, UAV and Data Link.

i. GCS Block

The GCS includes four subblocks as shown in Fig. 6: ControlCommandDevice, GcsProcessing, StatusDisplay and GcsTransceiver.

- Takeoff, Landing, starting and ending point, initial speed and change speed are mapping into ControlCommandDevice.
- Calculate distance, Time, Analysis UAV data are mapping into GcsProcessing.
- Showing the output data are mapping into StatusDisplay.
- Prepare data to send via data link, also get data from data link are mapping into GcsTranceiver.

ii. UAV Block

In the other side, The UAV includes seven subblocks: UAVPosition, SpeedSensing, AltiMeter, Camera, OnBoardProcessing and UavTransceiver.

- Get current location is mapping into UavPosition.
- Get current speed is mapping into SpeedSensing.
- Get current altitude over the sea level is mapping into AltiMeter.
- Get some pictures is mapping into Camera
- Generate control signal to steer UAV based on the control commands that received from GCS, manage data that get from the sensors are mapping into OnBoardProcessing.
- Prepare data to send via data link, also get data from data link are mapping into UavTransceiver

Fig. 7 illustrated the UAV architecture.

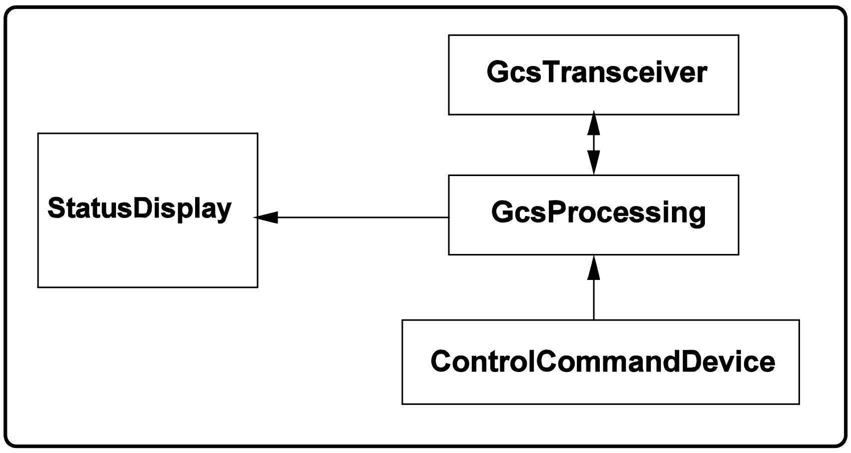

Fig. 6 GCS Architecture Model

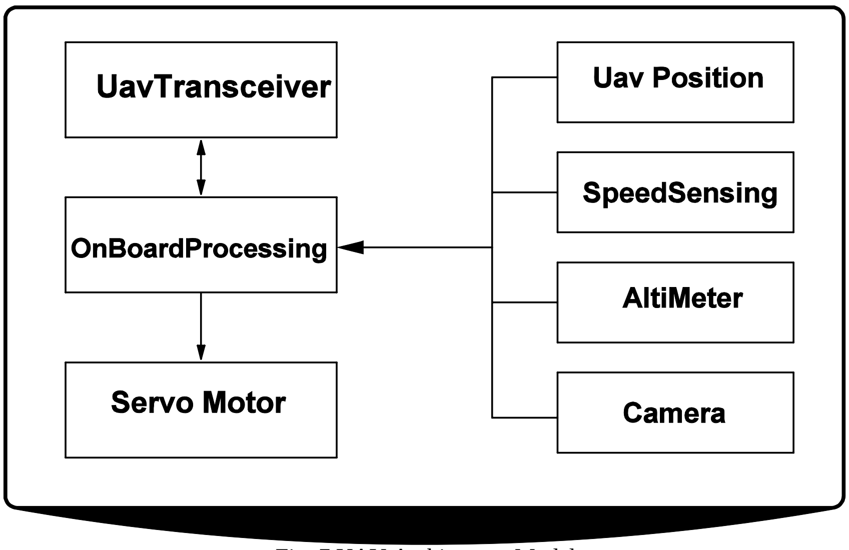

Fig. 7 UAV Architecture Model

iii. Data Link

This part represents the communications of the system and can be implemented by a variable.

2. HW/SW Partition

This is a very important step, HW/SW partition is achieved to get the better trade-off between the performance and cost. This system is partitioned into two parts as shown in Fig. 8.

i. Hardware

Includes the following blocks: UavPosition, SpeedSensing, AltiMeter, Camera and UavTransceiver.

ii. Software

It represents the program that will achieve processing task over the data that get from the sensors and data link.

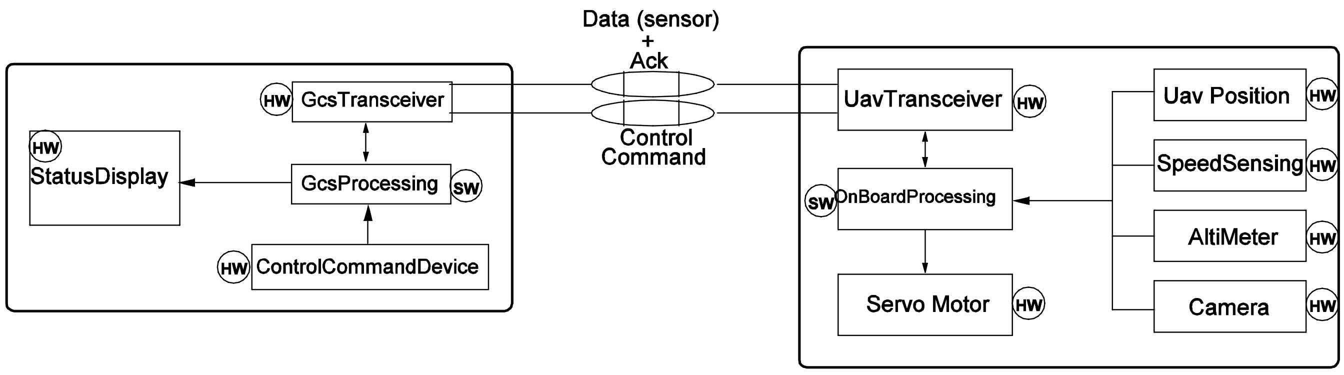

Fig. 8 Model after Partitioning

D.Architecture Exploration Phase

We decided to start architecture exploration at level 2 of the functionality that illustrated before in Fig. 5. The functionality will be allocated between two blocks.

- Ground Control Station (GCS) Block.
- Unmanned Aerial Vehicle (UAV) Block.

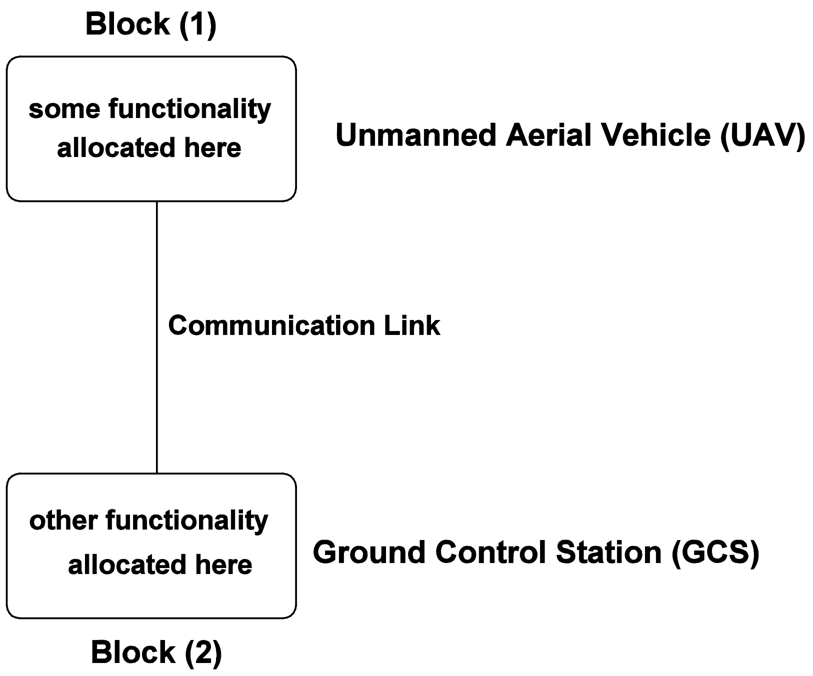
Block (1)
some functionality allocated here
Unmanned Aerial Vehicle (UAV)
Communication Link
other functionality allocated here
Ground Control Station (GCS)
Block (2)

Fig. 9 System Blocks

As mentioned before, the functionality (level 2) is target, navigation fly and take pictures. We proposed to distribute the functionality between GCS and UAV at this level as following:

1. Decide target is allocated to GCS.
2. Take some pictures allocated to UAV.
3. The navigation and fly can be allocated in GCS or UAV, so there are four combinations (four options) as shown in Fig. 10 and block diagram of the system with this option is illustrated in Fig. 11.

functionality / options	Target	Navigation	Fly	Take Picture
option_1	GCS	UAV	UAV	UAV
option_2	GCS	GCS	UAV	UAV
option_3	GCS	UAV	GCS	UAV
option_4	GCS	GCS	GCS	UAV

Fig. 10 Functionality Distributions between UAV and GCS Blocks

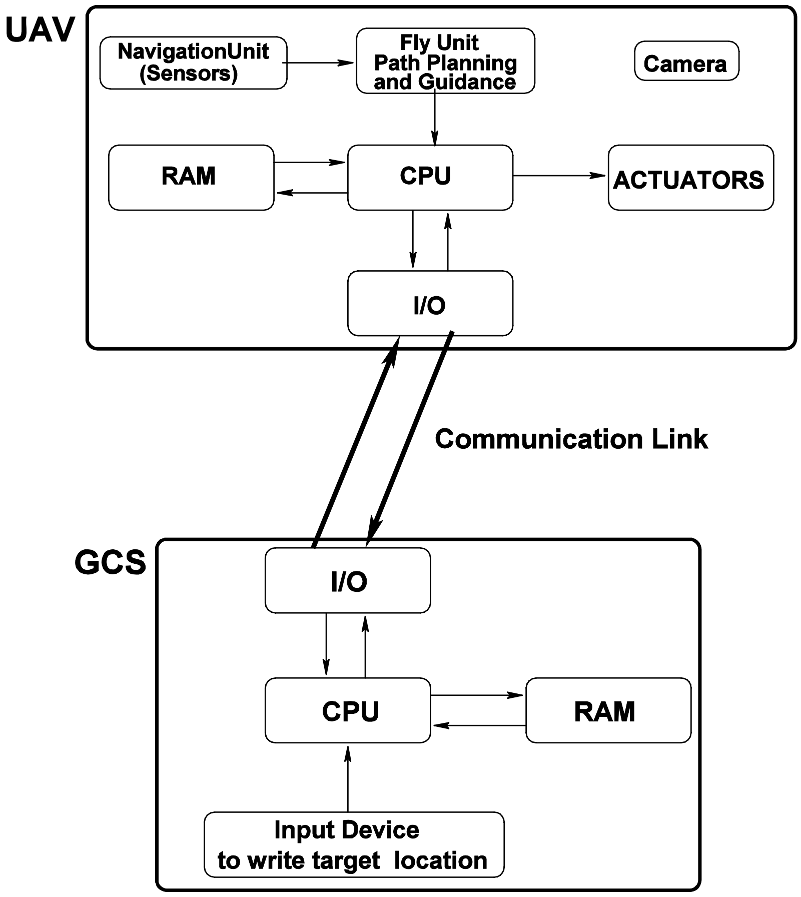

Fig. 11 Option 1: GCS (target) – UAV (take pictures, navigation, fly)

IV. Simulation of the UAV System

In this work, the modeled system is focusing only for the manual remote control system from the ground station. As we illustrated above, UAV system contains two subsystems, GSC and UAV. We have to model these subsystems concurrently. In the communication side, there is a wireless communication scheme between these subsystems. Also each subsystem involves significant communication internally (between its components). Moreover, each subsystems consists of many components that performing some function. All of the above phase must be achieved based on the wireless SystemC methodology [4].

The operational data are measured vehicle variables and control system variables. They are gathered and sent to the GCS. Those operational data comprising measured vehicle variables and control system variables. The measured vehicle variables including: altitude above sea level, vehicle speed, geographical position (latitude and longitude), actuators position, pitch rate - roll rate - yaw rate axes of the vehicle and accelerations in the (x,y,z) vehicle axes. The control system variables including: operational mode (automatic - manual), controllers state variables and error notifications. The system model that implemented at system level using SystemC is illustrated in Fig. 12.

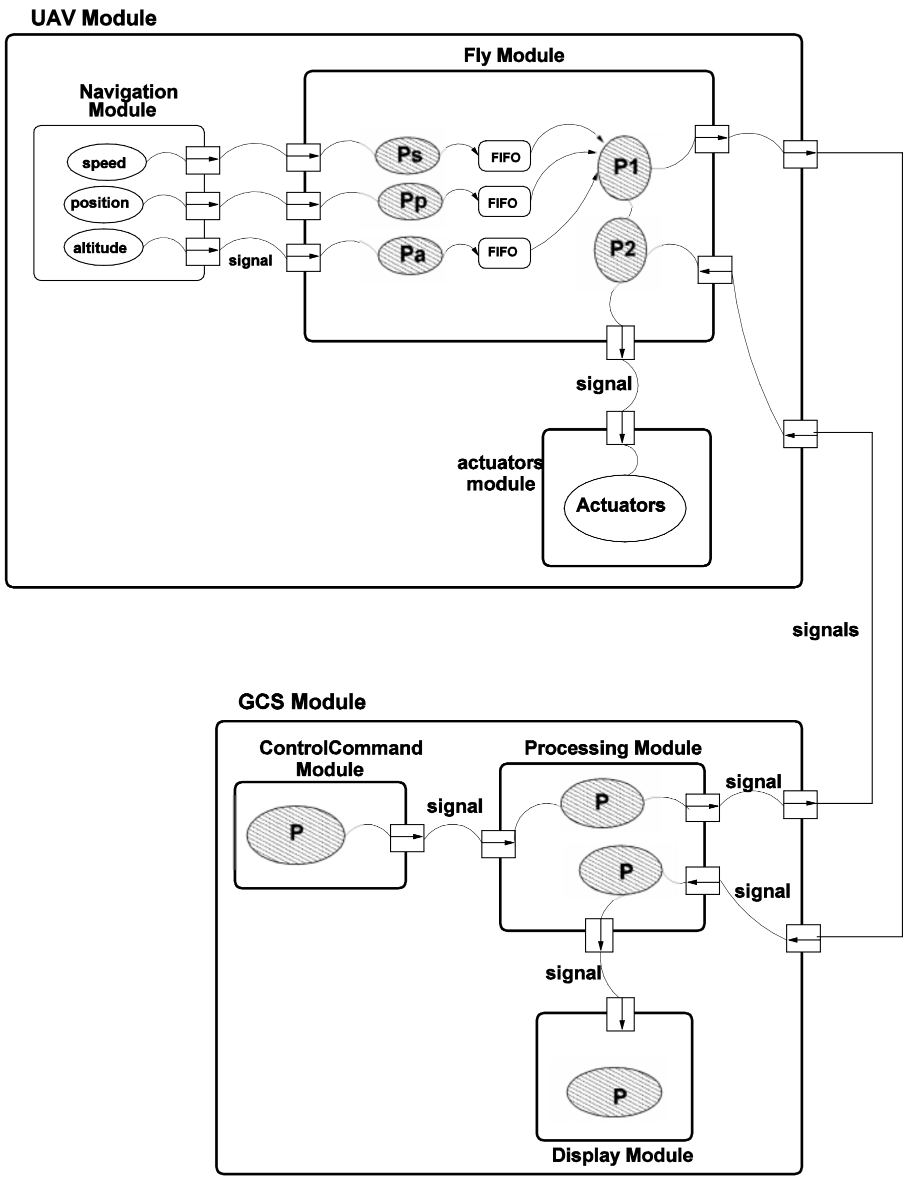

Fig. 12 UAV System Simulation - option 1

A. UAV System Program Structure

As mentioned above, the source code file structure of the UAV system is implemented using C++ and then refined to SystemC. The entire system consist of six implementation files and five header files (Fig. 13) as following:

- Gcs class: it represents Ground Control Station. It contains gcs.h and gcs.cpp.
- Uav class: This is an Unmanned Aerial Vehicle UAV. There are some functions implemented inside this class to represent sensors. It contains uav.h and uav.cpp.
- Points class: This class is used to encapsulate the points. It contains point.h and point.cpp
- Tools class: This class is used to find a suitable path between starting point and ending point and return it to GCS. Also it is used to calculate the distance based on the points locations. It contains tools.h and tools.cpp
- Uav_data class: This class is used to encapsulate the measured data that got from the sensor in Uav. The data such as velocity, current location and altitude. It contains uav_data.h and uav_data.cpp
- Driver.cpp : It is the driver of this application. It contains a main function.

The UAV System Sequence Diagram is shown in Fig. 14.

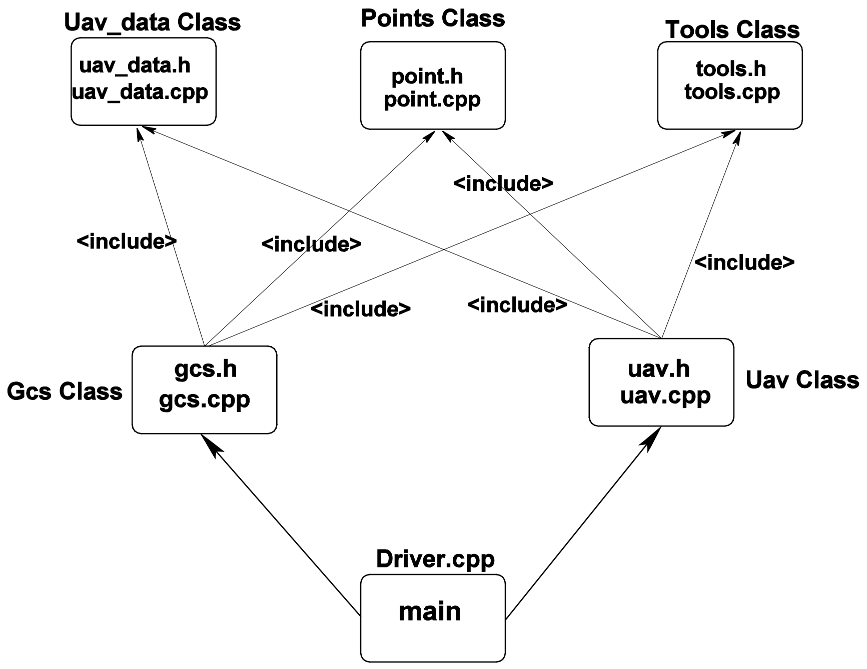

Fig. 13 UAV System Program Structure

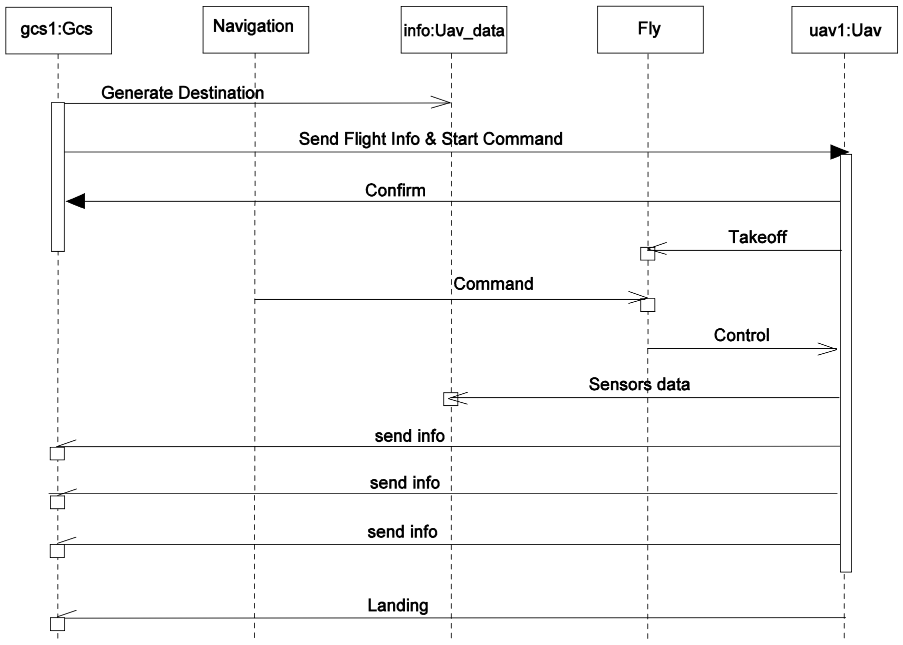

Fig. 14 UAV System Sequence Diagram

V. Conclusions

This work illustrates how to use SystemC wireless methodology to model complex wireless systems. As a result, the integration of communication modeling into design modelling is introduced in the early stages of system development. The design is developed by employing a system-level model of a noisy wireless communication channel [2] that fulfils SystemC language requirements in order to support wireless systems. We can then introduce some noise, communication delays and communication protocols.

In the last stage of this work, the SystemC wireless methodology is applied to the model of a UAV system that selected as a case study in order to how we can integrate communication modeling into the design modeling at early stages of the design flow. To our knowledge, this is the first time that the modeling of a UAV has been undertaken in SystemC and incorporated into a uniform design methodology, suitable for developing new technologies following the SoC design methodology. The final results of the modeled system have been validated and it has been proven that communication has a big impact in system dynamics, i.e., small changes in the wireless specifications create big changes in the system dynamics. As a future work, we will discuss how the system parameters will affect system behavior.

REFERENCES

[1] Aftab Ahmad. Data Communication Principles For Fixed and Wireless Networks. First edition, Jan. 2003.
[2] N. Ahmed, I. Aref, F. Rodriguez, and K. Elgaid. Wireless channel model based on soc design methodology. Fourth International Conference on Systems and Networks Communications (ICSNC), September 2009.
[3] Dan Brown and. A beginners guide to uml - part ii. Dunstan Thomas, 2009.
[4] Ibrahim Aref. Wireless Extension to the Existing SystemC Design Methodology. PhD thesis, School of Engineering, University of Glasgow, University of Glasgow, Glasgow, UK, February 2011.
[5] D. Black and J. Donovan. SystemC: From the Ground-up. Kluwer Academic Publishers, first edition, 2004.
[6] L. Cai, P. Kritzinger, M. Olivares, and D. Gajski. Top-down system level design methodology using specc, vcc and systemc. Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE), 2002.
[7] Reed Siefert Christiansen. DESIGN OF AN AUTOPILOT FOR SMALL UNMANNED AERIAL VEHICLES. PhD thesis, Brigham Young University, August 2004.
[8] G. Domer and P. Gajski. System Design: A Practical Guide with SpecC. Kluwer Academic Publishers, 2001.
[9] Donald Bell. Uml basics: The class diagram. ibm, September 2007.
[10] A. Ghosh, S. Tjiang, and R. Chandra. System modeling with SystemC. In ASIC, 2001.
[11] T. Groetker, S. Liao, G. Martin, and S. Swan. System Design with SystemC. 2002.
[12] KYUHO LEE. Development of unmanned aerial vehicle (uav) for wildlife surveillance. Master’s thesis, UNIVERSITY OF FLORIDA, 2004.
[13] R. Walstrom, J. Schneider, and D. Rover. Teaching system level design using specc and systemc. Proceedings of the IEEE International Conference on Microelectronic Systems Education (MSE), 2005.