A Mixed-Reality Environment for Digital Control Systems

K. Henke and H.-D. Wuttke
Technical University of Ilmenau /Department of Integrated Hardware and Software Systems, Ilmenau, Germany

***Abstract*— Learning the design of control systems requires not only deep theoretical but also practical experience that can be get in laboratory work. Goal of the projects "JGIFT" and "FIPS" - created by members of the Institute of Theoretical and Technical Computer Science of the Technical University of Ilmenau - is to examine and implement new techniques for a development and training system based on Finite State Machines (FSM). In our contribution we would like to present means and methods required for providing this tool set web wide for a large user base and independent from the operating system.**

Index Terms*— Remote engineering, Web-based education, Web-based design tools, Distance Learning, Remote Control via Internet

I. INTRODUCTION

A main teaching concept at the Department of "Integrated Hardware and Software Systems" of the Technical University of Ilmenau deals with the design of complex digital control systems. This concept includes the following lectures:

- Technical Computer Science
- Design of Digital Control Systems
- Design and Validation of Complex Parallel Systems

The lecture *Technical Computer Science* is held for undergraduate students in the first semester. It deals with the basics of Boolean algebra, combinational logic and simple sequential circuits. This course is supported by "Living Pictures", a collection of highly interactive Java applets [1].

Main topics in the lecture *Design of Digital Control Systems* are various minimization techniques for logical expressions, dynamic effects in combinational and sequential circuits and the design of digital control systems based on Finite State Machine (FSM) descriptions. This lecture is accompanied by a text book [2] and a set of tools, which students can use in a 4-hour laboratory [3].

Students in high-level courses have the possibility to deepen their knowledge in the lecture *Design and Validation of Complex Parallel Systems.* In this course the students learn different methods and tool concepts to design and validate complex design tasks. For a better understanding some accompanying practical designs are beneficially. Unfortunately such designs are too complex for a single lecture or a 4-hour laboratory. That's why we organize so called "project seminars" during a whole semester. The time for the project seminar is calculated with an amount of 2 hours a week.

Examples for such design tasks are e.g. controls for technical facilities like an elevator, a vending machine or a traffic light controller.

In our contribution we want to give an example of such a complex design task and demonstrate how the students can use different tools during several design steps remotely.

II. EXAMPLE

One of the technical facilities to be controlled is a hardware model of a four floor elevator, depicted schematically in Figure 1. The position of the elevator can be controlled by orders from outside and inside the elevator: Outside there are two buttons "up" and "down" at the second and third floor. At the first floor only one button "up" is available and at the fourth floor only a "down"-button can be pressed. Inside the elevator one button exists for each floor. All buttons can be illuminated to show that the button request is accepted by the control.

To determine the elevators position there are four sensors, one on each floor, which gives an active "1" signal, if the elevator has reached the floor.

The students are challenged to create a control that solves the following task:

"An elevator control algorithm has to be designed, which controls the model like a usual comfortable elevator in the following manner:

It is possible to press any button inside or outside of the elevator at any time. The elevator should move at the shortest way to the first requested floor. The elevator should stop during this move at any floor where a request in the same direction exists.

A pressed "up"-button at the third floor causes for instance that the elevator stops there on its way to the

fourth or an upper, but doesn't stop, if the down button is pressed.

To signalize that the control has been accepted a pressed button (or a request) the control algorithm should "switch on" the related button-light.

To move the elevator in the upper direction, signal y_u has to be set active "1" and the signal y_d has to be set active "0". To move the elevator in the opposite direction, the signals have to be set vice versa."

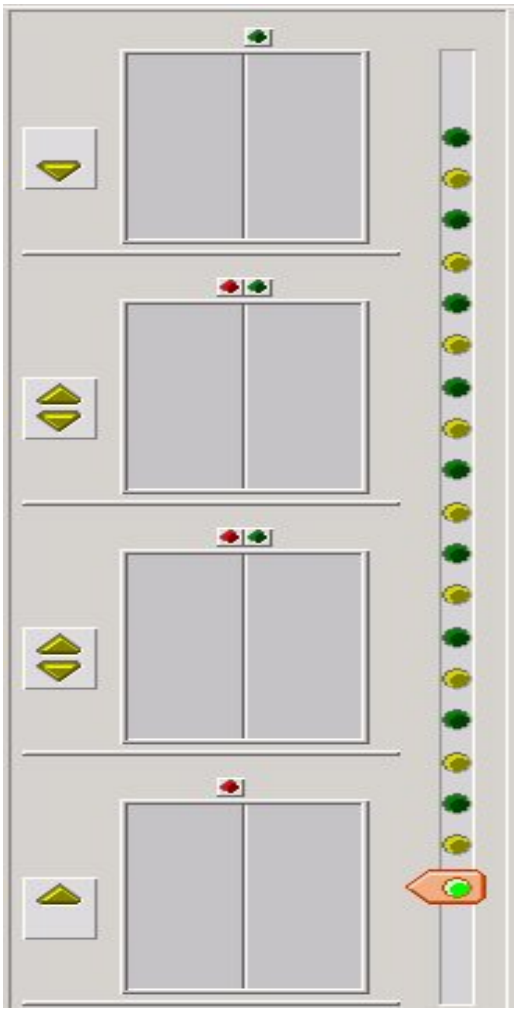

FIGURE 1: SCHEMA OF THE VIRTUAL ELEVATOR MODEL

To realize a solution for this task, the students can apply the learned methods, description techniques and tools. Because of the high complexity of this task, the work is organized in groups of 3 - 5 students.

The next chapters describe the design process and the usage of the tools during the single design steps. We show how parts of the design can be realized in parallel and how they can be composed. We also discuss the architecture of the developed remote laboratory and the organization of the group work and project administration.

III. THE DESIGN PROCESS

The design process of digital systems usually consists of the conceptual formulation and the design of the control so that the student can finally achieve a validated control. Figure 2 gives an overview of this process and describes the design phase more detailed.

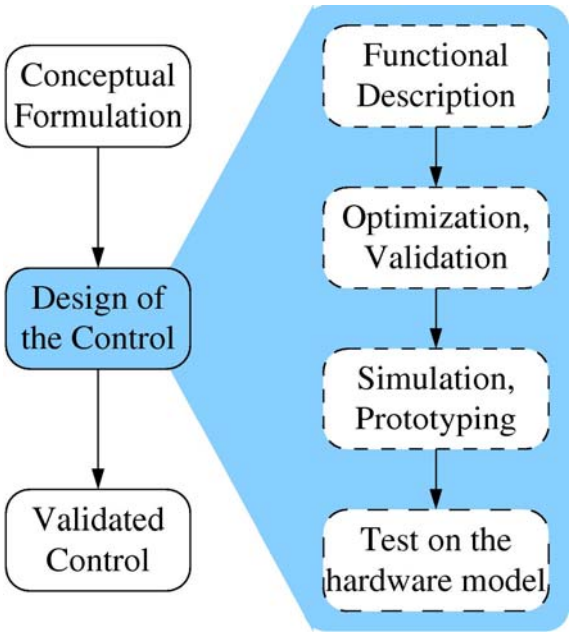

FIGURE 2: DESIGN PROCESS OF DIGITAL SYSTEMS

For the *functional description* we use parallel Finite State Machines as description technique [4].

This description method allows a formal *optimization*, *verification* and *validation* of the design.

A *simulation* and functional *prototyping* helps to find design errors. For instance, if one of the button lights is not handled by the control, this error can not be identified with formal methods. In the run-up to the practical, simulations and animations in "virtual worlds" are often used for verifying the developed solutions. The physical behaviour of the hardware model and its environment as well will be emulated (e.g. as simulation model). The user can influence his "**virtual world**" and analyze the reaction caused by his control algorithm. Dynamic activities will be animated and in some cases supported by corresponding sound effects.

These steps have to be executed in a loop until all errors are detected. An essential disadvantage of this method is that *real disruptive factors* (e.g. failure of single components, mechanical problems) can't be recognized by the underlying virtual environmental model. Mostly only a simulation of predetermined malfunctions is possible. Unconsidered sources of error lead to undetected failures of the control because the corresponding environmental situation was not simulated beforehand.

That's why the fault free design algorithm finally should be tested on a *real hardware model* (see Figure 3).

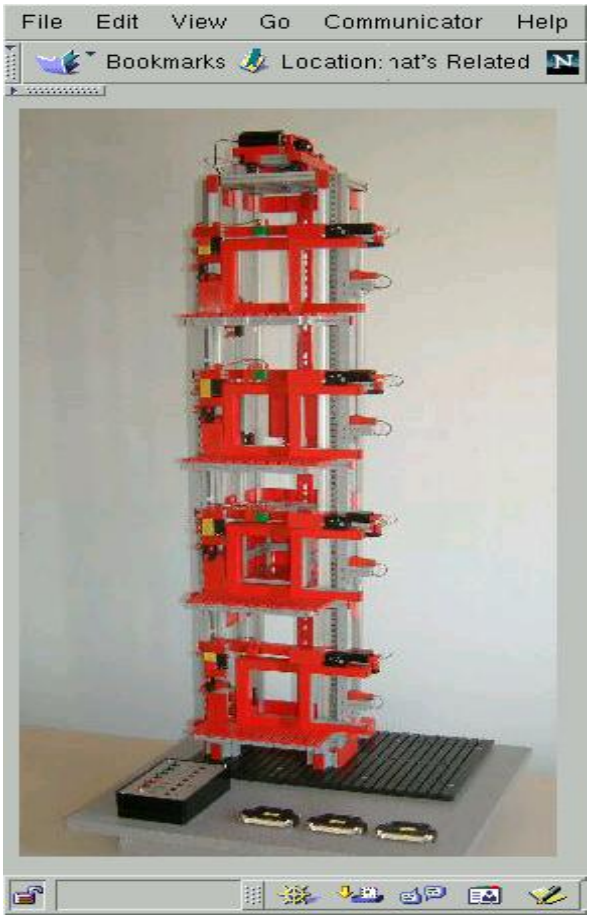

FIGURE 3: HARDWARE MODEL OF THE ELEVATOR

Assuming real laboratory conditions disturbances can appear causing the specialty of these situations. Including such real disruptive factors leads - in contrast to simulation - to a distinctly closer relation to practical conditions. Furthermore it should offer the students stimulus regarding the design of safety critical control systems. Additionally we'd like to concentrate on giving the student the chance to check his prepared control algorithms and to correct or modify the design.

IV. THE TOOL SET "JGIFT"

An example for a tool set supporting all the above mentioned design steps is the "JGIFT" system, developed at the Institute of Theoretical and Technical Computer Science of the Technical University of Ilmenau [3].

During their project courses students have the task to build a specification and to examine it based on a first formal verification and simulation. Theoretical foundation for this formal description is a tool internal description language for finite state machines ("FSMDL" - Finite State Machine Description Language), providing methods for general data administration, e.g.

- writing and reading of automata parameters,
- management of input and output variables,
- handling of edge weights and coding of nodes,
- execution of decomposition, composition and analysis,
- execution of verification, simulation und testing,
- connection of external program modules (e.g., model checker),
- connection of external hardware

for the designed automata. This description language is the basis for the tool set "JGIFT". The abbreviation "JGIFT" describes the objectives of this experimental system and stands for

- ***J***ava based
  - It means a web-wide usage by different users, which can solve their tasks on different platforms.
  - Complex systems often developed in teams using different computer platforms
  - Because of the chosen architecture as client/server system, the students can always use the most actual version of the JGIFT system
  - Changes and improvements regarding the JGIFT system have immediate influence on all users (e.g. enhancing the functionality by embedding new features or exchanging existing with improved algorithms)
- ***G***raphical
  - Graphical user interfaces always providing an intuitive basis for the work
- ***I***nteractive
  - JGIFT offers a high degree of interactive usage of different tools for decomposition, composition and analysis of digital control systems
  - The development steps can reach a high level of transparency. This means that the student can give (on single design steps) different alternatives a try and can analyze the effects.
- ***F***SM
  - JGIFT uses Finite State Machines (FSM) as theoretical basis.
- ***T***ools
  - A tool set, used in the *design process* of digital systems int different scenarios including FSM manipulation and validation.
  - *Administrative* tools for user and group maintenance, student's work place handling (project organization) and cooperative work of student groups.
  - Integration of external systems (e.g. model checker) and remote labs.

The "JGIFT" system offers different features. Visualization and animation allow observing and testing the properties of the design. In connection with formal design techniques simulation and prototyping are used

leading to a foundation for the development of a reliable system design during the first iterations of a cyclic design approach.

V. "JGIFT" IN THE DESIGN PROCESS

In the following we describe how students can use the "JGIFT" system to solve the above mentioned elevator design task. At first they have to develop a hierarchical concept for a system of automata. Each automaton should solve a special part of the whole control task so that the composition of these automata realizes the required control task. Figure 4 shows the structure of one possibility of such a hierarchical design.

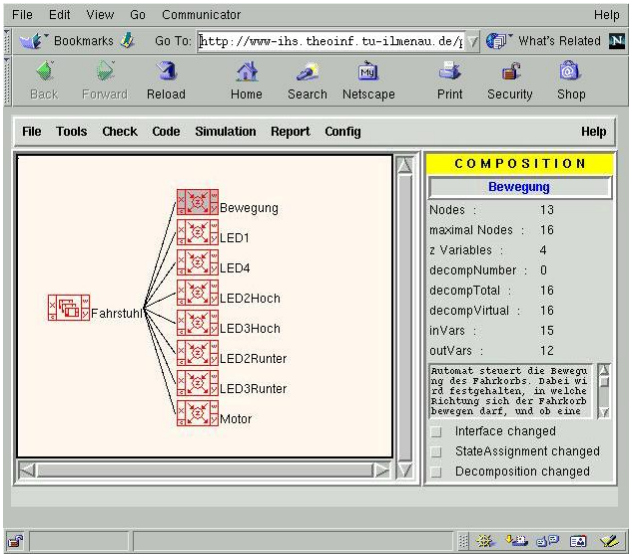

FIGURE 4: HIERARCHICAL DESIGN

The solution consists of eight automata which work in parallel. The first automaton (called "Bewegung") is responsible for the motion of the elevator. It gets information about the actual and the required position (floor) and finds the shortest way to the next requested floor. The next six automata handle the button lights and generate signals for the requests. The last automaton controls the engines.

The single automata can be specified in a graphical manner by using an automaton editor ("FSM-editor"). Figure 5 shows a screenshot of the editor and the graph of the first automaton.

If all the single automata are designed they can be composed by using the composition feature of the "JGIFT" system.

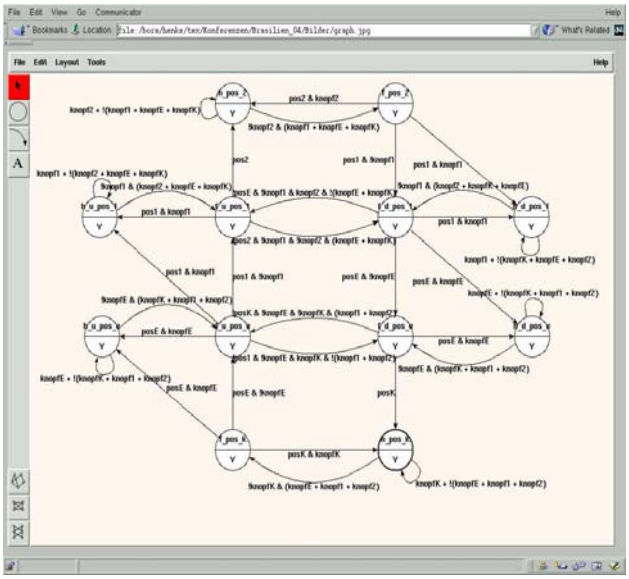

FIGURE 5: AUTOMATON GRAPH FOR THE AUTOMATON "BEWEGUNG"

To check the functionality of the whole design some special simulation features are included as integral part of the "JGIFT" system. This offers various possibilities for the execution of simulations, e.g.:

- generation of executable prototypes from the general design and parts of it as well
- possibilities for step by step and parallel execution of the prototypes,
- visualization of the simulation process at the specification level (e.g. FSM-editor),
- features for test pattern generation,
- linking of external hardware to simulation environment,
- possibilities for analyzing the simulation process offline afterwards,
- possibilities for code generation.

The user can interfere into the simulation process using graphical controls. That leads to appropriate state transitions caused by the changed set of input variables. Via waveform simulation – shown in Figure 6 - the temporal progression of input and output variables and the internal system variables of single automata can be shown. Also the behaviour can be visualized at the level of the system to be controlled, offering a first impression about the correct behaviour of the specified system. To do this there exist visual models of the systems to be controlled. They can be selected out of a visual model library.

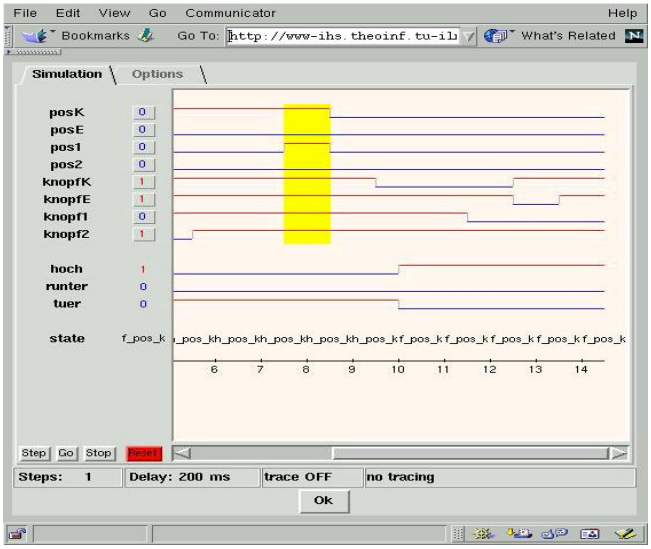

FIGURE 6: WAVEFORM SIMULATION FOR THE AUTOMATON "BEWEGUNG"

Until now suitable tool sets were available on special servers in a local network only and offered no web interface. We used them for students in high-level courses for individual studies. To teach an even bigger number of students it is necessary to split them into small groups and to organize special lectures in laboratory rooms.

Additionally for these purposes we have developed a web-interface to control the tools and the models remotely. By using this web-interface, the student is able to

- enter his created and already by the JGIFT system validated control algorithm,
- handle the laboratory procedure (start, stop, reset),
- change environmental variables, if necessary (e.g. before starting the experiment the elevator can be placed in any position interactively to analyze different special conditions) and
- watch the laboratory procedure by supervising the model and environmental variables in an I/O monitor or by observing the hardware model directly via Web-Cam.

At any time the student has the chance to correct his algorithm in case of faults. Therefore he is able to achieve a fault free solution (a validated control algorithm) step by step.

Figure 7 gives an overview of the system architecture. For more details see [5,6].

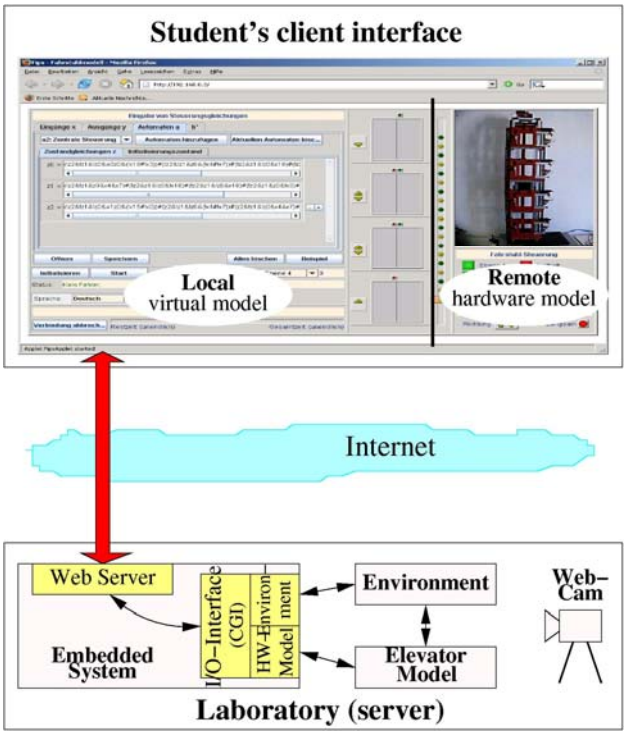

FIGURE 7:THE CLIENT-SERVER-ARCHITECTURE OF THE REMOTE LAB

VI. USER AND PROJECT MANAGEMENT

To offer a comfortable server based administration for users and their projects without having to deal with details of saving or version management, an administration database for users, user groups and their projects has been implemented additionally to the "JGIFT" server. For that reason the relational database *PostgreSQL* [8], a free SQL implementation is used.

By using this database it is possible - besides single user management - to allow *user groups* a joint (cooperative) work on one and the same project (e.g. design of parallel automata). Therefore the *user interface* (the central interface for the call of certain services) provides methods for user group management. They offer ways

- to *create* new groups by a user.

  This user becomes the administrator of the group automatically.
- to *join* an existing group.

  A new user will be managed as guest of the group at first (with limited rights). After clearance by the administrator the user is allowed to access and work with the group projects.

- to *leave* a group.
- to *delete* an entire group (only by the administrator).
- to *create* new group FSM projects.

For team work different users can access different parts of the FSM project (e.g. parallel automata).

Figure 8 describes the principle of the project and user management of the JGIFT project.

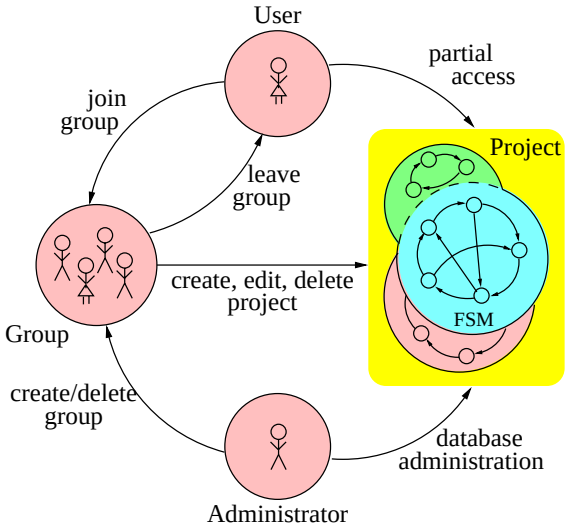

FIGURE 8: PROJECT AND USER MANAGEMENT OF THE JGIFT SYSTEM

To allow interchange between users working on a common project during a session a simple *chat system* was integrated.

ACKNOWLEDGMENT

.

The authors would also like to thank Jürgen Schmidt, Sven Hellbach and Friedemann Schoener for the inspiring discussions during the conception of the projects and the programming support during the implementation phase.

REFERENCES

[1] H.-D. Wuttke, K. Henke, "Teaching Digital Design with Tool - Oriented Learning Modules (Living Pictures)", *32nd ASEE/IEEE Frontiers in Education Conference*, Boston, Mass., November 6 - 9, 2002, pp. 25 – 30.

[2] H.-D. Wuttke, K. Henke, „Schaltsysteme -eine automatentheoretische Einführung", Pearson Studium, München, 2003.

[3] JGIFT, "Java-based Graphical Interactive FSM Tools", *Project Homepage*, http://www.tu-ilmenau.de/ihs/projekte/jgift, TU Ilmenau.

[4] J.E. Hopcroft, R. Motwani, J.D. Ullman, "Introduction to Automata Theory, Languages, and Computation", Addison-Wesley, 2001.

[5] K. Henke, H.-D. Wuttke, S. Hellbach, "Laboratory via Internet – New ways in education and research", *Int. Journal of Computers and Applications*, Vol. 25, No. 3, 2003, pp. 157-163.

[6] K. Henke, H.-D. Wuttke, "Web based educational tool access", *IASTED International Conference Computers and Advanced Technology in Education (CATE 2003)*, Rhodes, Greece, 30 June – 2 July 2003, Acta Press, pp. 531-536.

[7] MARE, "Joint Master Degree Programme Remote Engineering", *Project Homepage*, http://www.mare-project.org, 2005.

[8] REAL, "Remote and Applications Laboratory", *Project Homepage*, http://www.real-online.net, 2005.

AUTHORS

K. Henke is with the Technical University of Ilmenau, Faculty of Informatics and Automation, Department of Integrated Hardware and Software Systems, POB 10 05 65, Germany (e-mail: Karsten.henke@tu-ilmenau.de).

H.-D. Wuttke, is with the Technical University of Ilmenau, Faculty of Informatics and Automation, Department of Integrated Hardware and Software Systems, POB 10 05 65, Germany (e-mail: dieter.wuttke@tu-ilmenau.de).

This work is supported by the project "MARE - Joint Master Degree Programme Remote Engineering" by the European Commission within the program "ERASMUS - Joint Development of Study Programmes at intermediate and advanced level", Grant No 29298-IC-1-2004-1-AT-ERASMUS-EUC-1 [7] and the INTERREG IIIC Operation ENABLE – Project REAL – 2NW16 [8].