INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 05 MELBOURNE, AUGUST 15-18, 2005

A MODULAR DEMONSTRATION TOOL FOR ASSESSING AND DEVELOPING PRODUCT CONCEPTS PAST THE INITIAL IDEA IN MOBILE DEVICE DEVELOPMENT.

Roope Takala, Turo Keski-Jaskari

Abstract

Traditionally product ideas have been demonstrated using prototypes for technical proof of concept and design mock-ups and renderings for visual proof. More recently virtual prototyping has somewhat taken over some of these tasks. However, the ability to incorporate both physical and functional elements into product demonstrations is difficult due to the massive amount of work involved to getting even semi functional product demonstrations to a presentable level. We have tackled this problem by developing a modular demonstration platform for mobile devices. The system allows the reuse of modules that provide basic functionalities thus freeing valuable resources to concentrate on the really new issues in the product concept to be demonstrated. In addition to the platform we have developed a process for demonstrating and developing concept ideas from the initial concept to just short of the product integration phase where all the product systems need to be essentially verified. Our method includes firstly the process of building the first product demonstration. Secondly the use of test methods and apparatus that has been adapted for use with the modular platform and thirdly the ability to evolve the demo device during the development iterations to finally include almost all of the systems and technologies of the final product. The modular design of the system has enabled, among other benefits, the move into multi-site development activities where all the sites have a common target but can still work in separate physical modules. When an individual module completes a design iteration a copy of it can be sent to other sites to be used as a part of the whole setup.

Keywords: Product Concept Development, Fuzzy font end, concept demonstrators.

1. Introduction

In a world with constant need for more efficient R&D continuous developments to improve productivity are a must. We have looked at this issue from the point of view of evaluating and developing new product features. Our focus has mainly been in features for the user interface of mobile devices, but the work also yielded benefits to physical testing of user interface components. [1]

The objectives of the research was to show that it is possible to develop a modular platform for product concept demonstration and its' use provides a possibility to increase the productivity of evaluating and developing product concepts and technologies. Additionally the objective was to develop the platform to be scalable enough to allow the iterative development rounds of concept technologies from initial trials to just short of final product integration. The target was also to drive the ability to do true and effective multi-site hardware development for the verification of product ideas and technologies

The problem was tackled by designing a module structure based on incorporating the basic hardware components required in mobile devices into HW modules that can be assembled together rapidly. The boundaries to variable modules were also clearly defined and to allow easy plug-in of new modules. Defining the boundaries of different modules in a way that allows the development of the modules separately, but still incorporates the possibility to test them with the entire system as soon as there is something to test. The clear modularization and boundary specification allowed scalability by enabling the development of the modules separately while still preserving an ability to work with the whole system. This approach also enabled the splitting of development of individual modules to different sites. As a result the best teams could be assembled to develop each module further.

2. Demonstrating process and demonstration platforms

A demonstrator is a device that has feature and design resemblance to the final product or product feature being demonstrated. The demonstrator does not have to have any physical or hardware components that could be used in the final product. This is what differentiates demonstrators from prototypes. Prototypes are used to test the real parts in a real kind of setup, to find out how the parts interact with each other and their surroundings. Aim is to detect any flaws in hardware or software design to have them corrected for the real product. [2]

When comparing to a prototype a demonstrator is closer to the end-user, it is a device that has the functionality and a design that resembles a ready product. It shows to the tester, test subject, or customer what it would be like if such feature would exist in a real product. The demonstrator is normally used to determine how potential user groups react to the features it incorporates. These new or new packaging of known features are what incorporates product concepts. Thus the demonstrator is ideal for evaluating the real usability and merits of new product concepts. [3]

The need for working demonstrators is especially visible in the field of mobile products, where the lifetime of products is short, and design cycles must be faster all the time. Demonstrators can be used to pre-evaluate the usability and feasibility of new product concepts and ideas before going into expensive and resource-hungry prototyping phase. Demonstrator can already show if a new idea cannot be useful at all, or what to change to make it useful.

When a demonstration platform is designed in a modular fashion it enables a step-by-step process of moving from very simple proofs of concepts to almost full-blown product prototypes, figure 1. In the process the earlier simple demos are amended by new modules and become parts of a larger system. This enables demonstrators to be utilized in various phases of the product development, to ease the moving new features and technologies to the prototyping or product design. For example, all new features originate from an idea. The inventor needs somehow explain the idea and its usage to others, for example by drawings on the back of a cigarette pack. If the idea sounds good and realizable, the process can continue with something that is already closer to reality, for example a wood mockup that would have some imitations of the new idea. At this phase it is normally a non-functional mockup showing mainly the shape and size of the device, some mechanically moving parts at most. The next step is to add some basic functionality this is where the first pre-determined demonstrator platform modules are taken into use. As the development process moves forward these simple feature demos can be integrated to a processor unit that runs product type SW. At this stage typically product SW can already be implemented and tested. The final step is to start replacing the demonstrator platform modules with integrated product hardware.

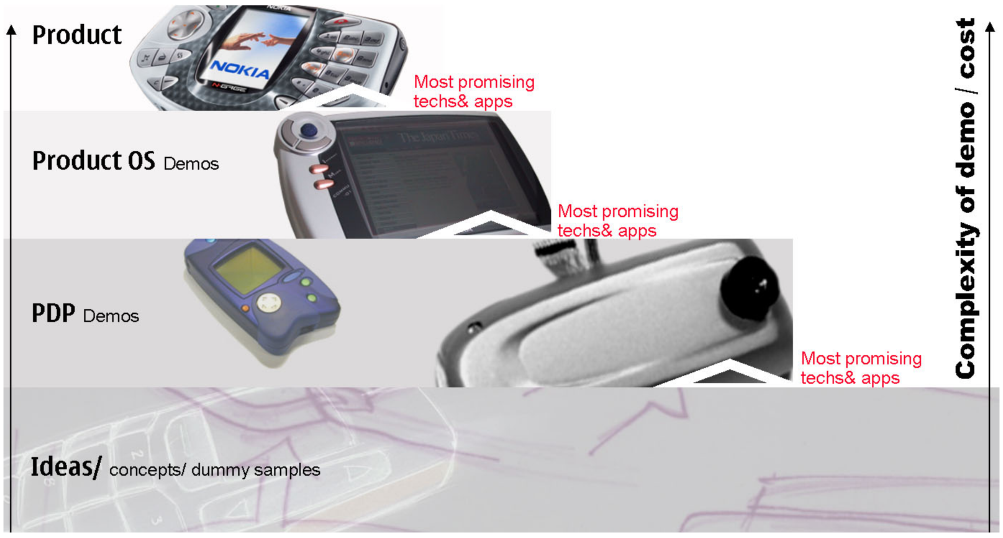
Product
Most promising techs& apps
Product OS Demos
Most promising techs& apps
PDP Demos
Most promising techs& apps
Ideas/ concepts/ dummy samples
Complexity of demo / cost

Figure 1: The step-by-step process for increasingly complex demonstrators utilizing the modular platform.

In product design environments there is usually neither time nor the resources to do everything from scratch, the ground is set for generating a demonstration platform. At its best, a well-designed demonstration platform can hasten and facilitate the whole process of making more demonstrators, and eventually evaluate larger numbers of product concepts more efficiently at smaller cost.

Product platform is “a set of subsystems and interfaces that form a common structure from which a stream of derivative products can be efficiently developed and produced.” [4] To generate a powerful and reusable demonstration platform, the requirements of different user groups and product types must be defined. Out of these the highest common denominator of the required features is selected to form a basic functional module that can be used as an engine for further demonstrators. Issues such as physical size and power consumption requirements of the features must also be taken into account in the selection process. After the main features needed are integrated into one core module, the generation of derivative concept demonstrators becomes easy and fast, enabling the developer to concentrate onto the essentials of each demonstrator case. [2]

Modularity is normally not a binary variable, so that every product family could be defined to be either modular or not modular. Ulrich & Tung [5] state two characteristics of a design that describe the degree of modularity:
1. Similarity between the physical and functional architecture of the design
2. Minimization of incidental interactions between physical components.
Thus the degree of modularity is higher when the modules are split according to their respective functions and when the interactions between them happen through defined channels and methods. And vice-versa, less modularity is present if the functions of the modules coincide and there is interaction between them that is not pre-agreed and well defined. In a real product this could be for example heat generated by one module, that

another module cannot withstand. Larses and Blackenfelt [6] also agree on dividing the modules by their functions, but further evaluate the meaning of today's trends of embedded software and the fading of borders between physical and functional modules. They also present the concept of strategic relations between modules. Strategic selection of what to include in certain module takes into account new criteria, such as buy vs. make and reuse vs. develop. All of this shows how important it is to define the whole modular structure proficiently.

As the platform for various demonstrations needs to be usable for developers in broad range of skills and competences, and for varying technical needs, defining and documenting the modular boundaries, e.g. interfaces offered for the users become a crucial issue.

A platform for product concept demonstrations needs to serve product developers with various and different levels of skills, and thus needs to have very clearly defined user interfaces. Besides offering the suitable interfaces for the demonstrator maker, the platform should also take into account the interfaces that could be usable in the next phase, e.g. prototype or even final product. This would facilitate the process of a real step-by-step process described earlier, and result also in financial advantage from re-use of components.

The challenge in designing the interfaces and setting the characteristics and limitations is not to rule possibilities out from future devices, but still keeping the interfaces reasonably simple. Additionally, the design should promote the users to use the same interface usage patterns for the new needs as with the earlier ones without restricting the applications. If an interface cannot support the user's new needs, or if it otherwise much easier to do what is needed bypassing the guidelines, it will inevitably happen some day. This will result in an increase of incidental interconnections, that Ulrich & Tung's second measure of modularity advises to minimize. To prevent this from happening, all the information needed to use the interfaces need to be available for the user, and so clearly defined that there is no need and no reason to try to cut corners anywhere. To help with the problems during the transitions from earlier platform to a newer revisions, Lehtonen & co [7] present the concept of dynamic modularisation (Dymo). Dymo defines a system level architecture according to business needs, and it serves as a framework for the subsystems that can be generated. This approach has been kept in mind to enable the re-use of modules developed for previous versions of the platform.

3. The Tool for Concept testing and technology verification

In a development environment people generate different kinds of mock-ups, demonstrators, and prototypes around the organization, or even in different organizations. This makes it challenging to learn from earlier mistakes and spread the knowledge from project to project over time and between working units. A well-designed platform should also ease up the concurrent development of demonstrators between several development sites.

The Product Demonstration Platform –project (PDP) has been on-going for several years, and the platform itself has gone through several iterations regarding internal parts and also the interfaces offered for the users. The latest revision, namely PDP 4.0, has now stabilized the module boundaries to level where the objectives set at the beginning have been met. The main one of these was to enable the completion of a concept demo in a matter of weeks not months.

The PDP platform consists of several modules. These can be split roughly op in three categories hardware (HW), software (SW) and mechanics. Although, it should be noted that some of the modules are a mix of these. The main HW module is the PDP 4.0 engine, which is sometimes, referred to as 'the platform' it is a motherboard that is the heart of the system. Other HW modules include user interface (UI) boards of different complexity and boards that contain sensors. The SW modules contain drivers for different devices that can be connected to the system such as displays and the previously mentioned sensors. Different SW UI's are also included in the SW modules. The mechanics module vary the most as they are dependent on the form and design of the demos, as a result they range from designed generic phone covers as shown later in figure 4, to off the shelf electronics boxes or reused covers from existing products such as mobile phones or PDA's.

To serve a broad range of users, PDP needs to be as flexible as possible regarding its electronics, mechanics and software. In core electronics this is handled by offering both C-programmable micro controller, and programmable logic device (PLD). The combination of a chip executing higher level language and a chip that offers means to effectively create hardware logic by a descriptive language give the user much possibilities to tackle with different types of hardware modules. Another point to note from hardware flexibility is power handling and offering, as different hardware modules may need to operate at different voltage levels. Thus PDP can output power from 4 different sources: directly from battery, regulated 3.3V, 1.8V and a voltage that can be controlled by software in between 1.5 – 5.5V.

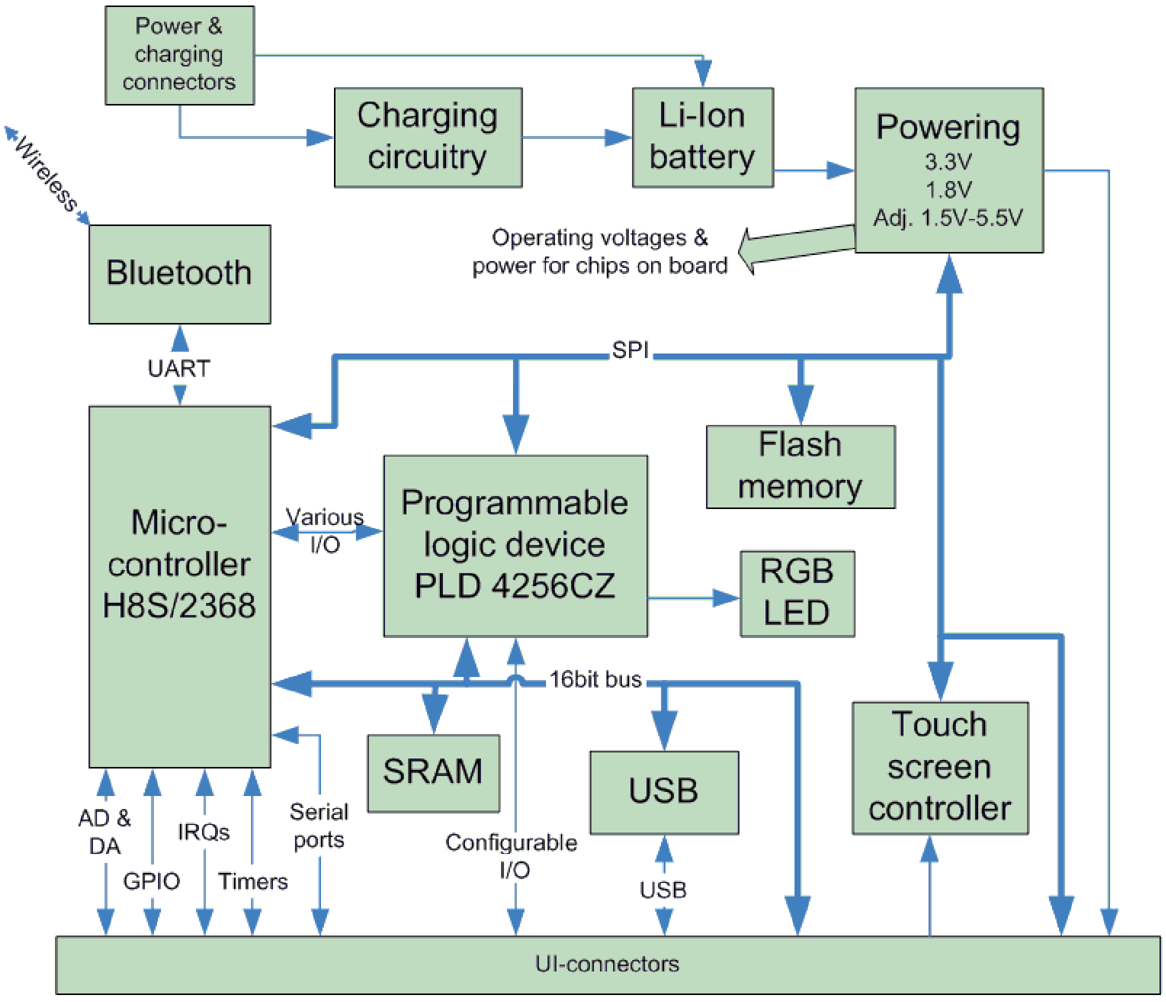

Figure 2: Block diagram of PDP 4.0.

The interfaces offered for the user are visible in the simplified block diagram of PDP 4 internal connections shown in figure 2. There are both parallel and serial bus interfaces, analog channels, general purpose input/output pins, reader for touch pads, interrupt lines, timers, and USB, Bluetooth and serial ports for the host connection. The UI-connector, which is the main interface to most of the developers, is of a type that does not require a special connector. Instead it incorporates simple pin connectors or even bare wires that can be found in virtually every lab or workshop. These properties make the PDP 4.0 engine, in figure 3, easily and immediately usable without the need for waiting for specialty parts to be ordered.

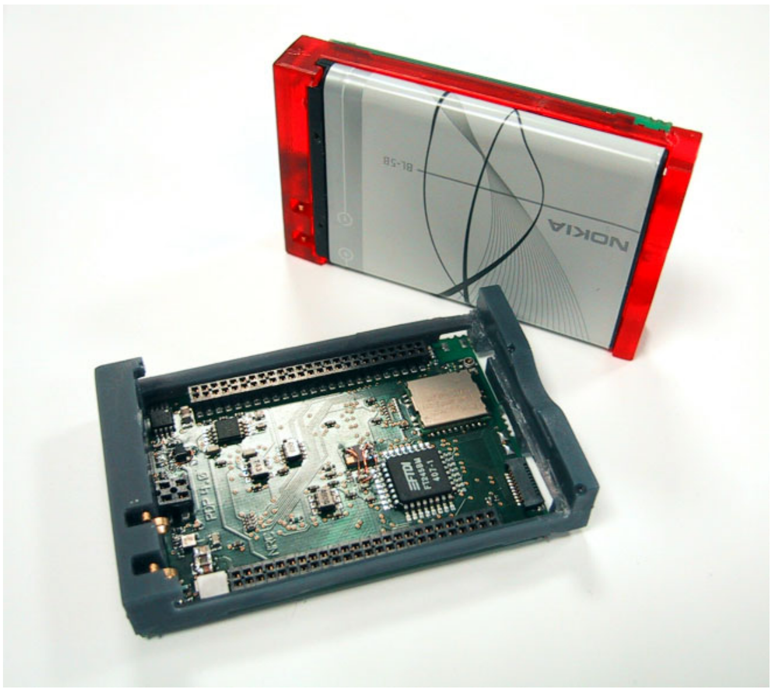
Figure 3: PDP 4.0 and battery in the default holder

In the software there must also be a clearly defined “platform”, e.g. well documented structure, basic set of low and higher level drivers, basic UI’s and stable and efficient interfaces to where new drivers and applications can easily rely. Different people can thus extend reusable software library all the time with drivers, new UI concepts or test programs to show and test their own new product feature concepts in practice.

The third aspect of modularity of the PDP 4.0 engine is its mechanical interfaces. It is an independent, small, and clearly defined module, that can be plugged to UI-boards with specified connectors. There are two 50-pin electrical connectors, and a power connector to interface own circuit boards and components, and a default plastic holder for the default type of li-ion battery. User can select to use the whole package with battery, or take just the PDP 4.0 engine itself and route power from elsewhere. The other mechanical modules are generally demonstrator specific.

4. Using PDP 4.0 for Feature Concept and Durability Verification of Touch pads

The PDP 4.0 has shown itself to be more versatile than anticipated in the beginning of the work. In the two use cases of the system the first one is a demonstrator of a product feature enabling pen input of characters. This is the type of use the platform was originally planned for. The second case is the use of the platform to automate a touch pad durability analyzer, for which the platform and the process philosophy of evolving design proved to be extremely suitable.

4.1 Touch pad UI demo with PDP

There are numerous demonstrators already made with different revisions of the PDP. One example of the user experience testing concepts is a device made for touch pad usage, seen in Figure 4. There the PDP's parallel bus interface is used to connect a color display, and serial peripheral interface to interface with a touch pad reader chip. The demonstrator is also connected to a PC computer through normal serial port. All the parts are put into covers that are made by silicon molding in a normal laboratory workshop. For the test users the device feels and roughly seems like a real product, but instead of phone electronics it has a PDP and several user interface modules, such as the touch pad, inside. Many of the parts are off the shelf standard components, but for example the display is a production model Nokia 7650 phone's display module, just interfaced with and controlled by the PDP software. There is significant freedom to use certain modules from existing products among the newly developed modules of the demonstrator platform.

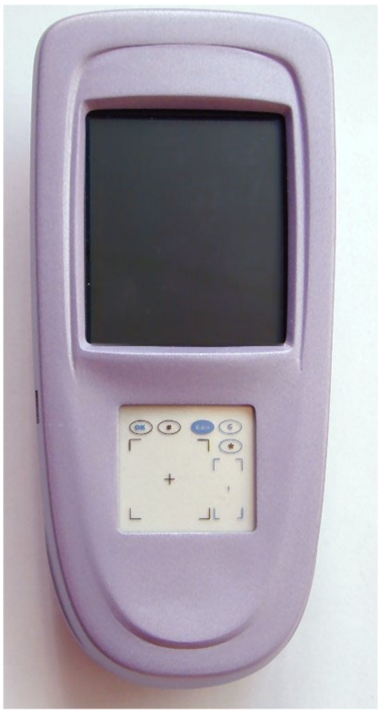
Figure 4: Touch pad usability concept demonstrator.

Typical user test situation is for example evaluation of the usability of Chinese handwriting recognition software in a real usage environment and in using a real hand-held device. In the test setup the user holds the device in one hand and writes Chinese characters to the touch pad. The drawn lines are transferred as coordinates to PC software, and also shown to the user as lines in the display. After a character is drawn, the recognition software is used to identify the characters and send the result back to the demonstrator, which shows the result to the user. Thus the functionality of the recognition software is transferred to a hand-held device imitating a real product. This enables actual numerical measurements from the usage of the device. These may include measures such as error rate in different usage situations, and times taken to draw the characters. These in turn help the development of the concept features further.

The touch pad UI demonstrator proved the initial assumption of a speedier development process. According to the project manager of the demonstration project, it required approximately four person weeks of work to complete the physical demonstrator. The estimation for completing a similar demonstration without the platform is in the order of three to four person months. In addition to this it gave indications that the other initial assumption that multi-site work would be facilitated. The touch pad UI demonstrator was developed on four sites in parallel, with one site in charge of the system and integration, another of the touch pad module, the third worked on the SW and the fourth site was in charge of the user testing of the system.

4.2 Touch pad analyzer automation with PDP

Main purpose of PDP platform is to work as an engine for various types of user interaction demonstrators, but as it offers a broad range of standard and documented interfaces, it can also be useful building block for physical verification, measurement, test devices and systems. The following is an example of how the PDP engine was used to evolve a test system that measures the rate of wearing of touch pads by adding automation to it.

The system originally consisted of a robot that wears down touch pads, and touch pad reader circuit connected through the PDP to a PC. Here PDP was only an interface module between a reader circuit and the PC. The system involved an operator too much to run the equipment continuously. A decision to upgrade the system was made. The upgrade should at least fulfill the following requirements:

- Wear and measure a number of touch pads at the same time
- Control the process automatically without operator for example during nights, weekends.
- Store the results to a set of clearly named files into the system, to the PC or to a network drive
- Measure the pads and filter out disturbance exerted by the system or environment
- Keep the operator informed of the progress and problems
- Easily configurable regarding different pads, measurement readouts, and wearing patterns / durations
- Provide easy way to connect pads with different physical interfaces
- Provide various interfaces for operator to configure and control the system (Serial port, USB, Bluetooth, Ethernet, M2M GSM/GPRS)

To meet up with the set requirements, it was decided that the master of the system is the PDP. Having PDP as the master allows for the PC with the user setup interface to be detached from the system during operation, and only connected back after the tests has been done.

The resulting system, the block diagram of which is shown in figure 5, still uses the robot, which wears down the touch pads using constant weights and wear methods, and a PC that logs the input data and offers the visual user interface for operator. The PDP runs a state-machine that schedules the operations and commands the other parties. PDP uses separate touch screen controller (TSC) –modules to get the actual coordinate readings from the touch pads. Because the original serial peripheral interface (SPI) would have needed different chip select for every TSC-module, an addressing system for the SPI-bus was developed. It can support up to 254 different touch pad addresses along the same cable. PDP is powered normally from an external power source, but can switch on the fly to use it’s own battery, in case of a power failure. Additionally the system can have Machine-to-Machine (M2M) interface module to inform the operator of the system status. [2]

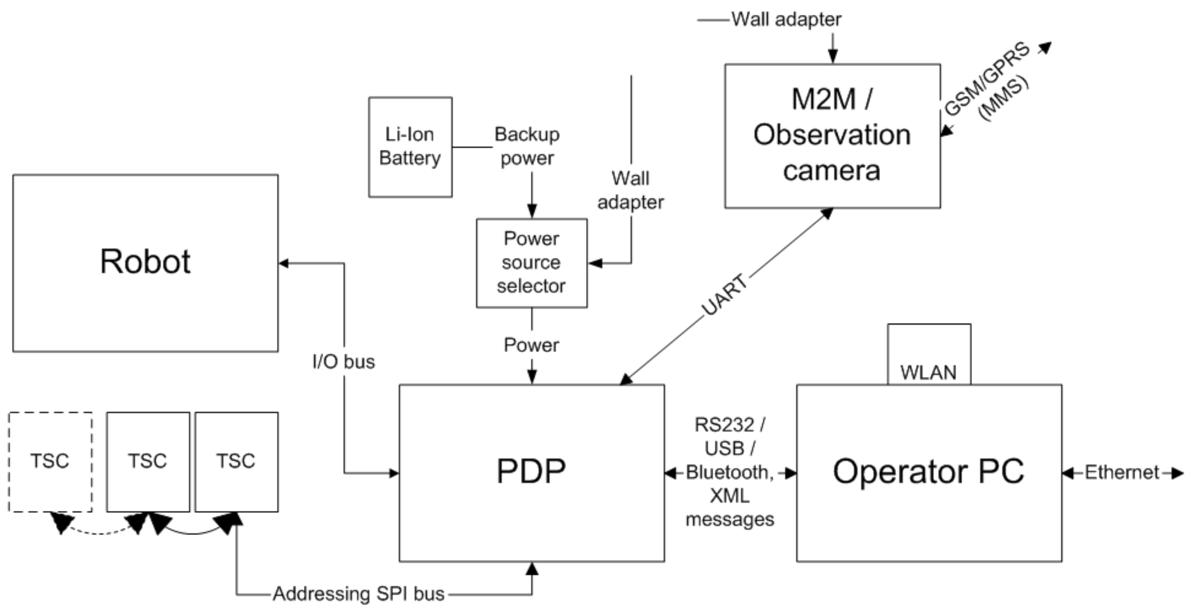

Figure 5: Touch pad measurement platform block diagram.

PDP can send the measurements directly (or only buffering some amounts) to the PC, or it can log the measurement data into its internal RAM and flash memories. When readouts are sent to the PC, PDP can keep a backup record in its memory.

Several types of interfaces offered by the PDP are used in the system. The graphical user interface is shown in the operator PC's screen, and after the setting up of the system is ready, configuration data is sent to the PDP over serial port in one file. XML structured setup file is also readable and modifiable by a normal text editor. Communication interface for the TSC-modules is the above-mentioned addressing SPI. Communication with the wearing robot uses general-purpose I/O pins to form binary coded decimal (BCD) numbers. By using octal numbers, only three pins + clock signal were needed to both directions. [2]

The new setup increasing the PDP's role has allowed test runs to be run up to four days without direct operator supervision. The use of the system autonomously during weekends and nights significantly increases the throughput of testing.

5. Conclusions

Based on the experiences from building the modular platform for demonstrating product concepts it can be concluded that by careful analysis of the types of product in question and planning of the module boundaries a platform can be created. Other similar approaches have generally made use of miniature computers such as Jappla, which is used to demonstrate future terminals in the Japanese IPV6 promotion program [8] or Gumstix [9]. The experience the authors have gained outside of this study from using matchbox sized computers clearly indicated that an intermediary system without an operating system and the subsequent restrictions on design and the workload of driver creation is advantageous in the early demonstration of product concepts and new product features. The concept demonstrators made on miniature computing platforms tend to be developed much further than the demonstrators the PDP is aimed for. So even though there are apparently other approaches to concept and product feature demonstration they tend to be either simpler or targeted at concepts developed significantly further.

The use of the platform has also shown that utilizing modular development tools can attain significant enhancements in the productivity of developing concept ideas. Specifically these enhancements have been seen in the easier and more parallel development and the speeding up of concept feature testing. Based on an initial review of about 20 projects using the system the platform has been calculated by the authors to reduce the development time of a basic demonstrator of a product concept from about 18 person months to just over three person months. Some individual demonstrations have been developed in a matter of weeks. Additionally the re-use of the modules has been significant. It must be noted though, that the PDP demonstration is most effective when demonstrating a few specific product features. The system agility in full multi-feature concept demonstrations tends to converge with the miniature computer platforms.

Based on our experiences it seems that a modular platform approach can be utilized effectively and with success in the process developing and evaluating concept features in the domain of mobile devices.

The future of the platform development work is currently open. One possibility for the future could be a "micro-PDP" (uPDP), a tiny multi-chip-module, which would integrate all or most of the PDS's functions into one BGA package. The internal logical structure of the uPDP would be compatible with the tabletop version. This would allow all the software that has been developed in PDP 4 to be directly uploaded also into the small uPDP. This enables testing of parts easily with PDP 4, and also a very compact final solution with the uPDP module plugged into an application board. Regardless of future development tracks the authors intend to use the PDP4 platform to develop new product concepts and test product features. The platform will also be made available to selected universities and research institutes to enable and promote their work with technologies and product features of mobile devices. A need for an upgrade of the PDP 4.0 platform HW is expected within two to three years.

Acknowledgements:

The authors would like to thank the colleagues that helped make the PDP, most importantly Mr. Jukka Jalkanen. Additionally we would like to thank Dr. Matti Ryynänen for the contributions in the cases presented.

References:

[1] Takala, R., Ekman, K., A “Typology of Product Concept Creation and Evaluation”, proceedings of Nord Design 2002, Trondheim, Norway, 2002, 315-324.

[2] Keski-Jaskari T., “Modular Platform for Testing and Demonstration of Handheld electronics devices”, Thesis, Helsinki University of Technology, Helsinki, 2004.

[3] Takala, R., “Product Demonstrator: A system for up-front testing of user related product features”, Journal of Engineering Design 2:2005.

[4] Meyer, M, Lehnerd, A., “The Power of Product Platforms”, The Free Press, New York, 1997.

[5] Ulrich, K, Tung, K., “Fundamentals of product modularity”, DE-Vol. 39, Issues in design Manufacture/Integration, ASME, USA, 1991, 73-79.

[6] Larses, O, Blackenfelt, M.,. “Relational reasoning supported by quantitative methods for product modularisation”, proceedings of the 14th International Conference on Engineering Design, Folkeson et al., The Design Society, Stockholm, 2003, 347-348.

[7] Lehtonen, T., Juuti, T., Pulkkinen, A., Riitahuhta, A., “Dynamic modularisation – a challenge for design process and product architecture”, proceedings of the 14th International Conference on Engineering Design, Folkeson et al., The Design Society, Stockholm, 2003, 339-340.

[8] IPv6 terminal demonstrator at the Net.Liferium 2003 exibition, http://k-tai.impress.co.jp/cda/article/news_toppage/13214.html, (27.5.2005)

[9] Gumstix inc., http://www.gumstix.com/, (27.5.2005).

For more information please contact:

Roope Takala, Nokia Research Center, Helsinki, Finland, Email: roope.takala@nokia.com

Turo Keski-Jaskari, Nokia Research Center, Helsinki, Finland, Email: turo.keski-jaskari@nokia.com