Universal Controller Module (UCoM) - component of a modular concept in robotic systems

K. Regenstein, T. Kerscher, C. Birkenhofer T. Asfour, M. Zöllner, R. Dillmann
Research Center for Information Technologies (FZI)
Interactive Diagnosis- and Servicesystems (IDS)
Karlsruhe, Germany
Email: regenstein@fzi.de, kerscher@fzi.de, birkenhofer@fzi.de, asfour@fzi.de, zoellner@fzi.de, dillmann@fzi.de

***Abstract*— At our institute we built a variety of robots. As these robots are quite different as well in size, shape and in actuation principle it would be very time consuming and inefficient to build a computer and hardware architecture especially tailored to the specific robot. In this paper it will be described how common aspects in robot control can be identified and how a modular software framework and a respective computer architecture can be mapped to modular components on the hardware side. A decentralized computer architecture based on embedded PC systems connected to local controller modules via CAN-Bus was developed. The requirements and restrictions that led to the development of these controller modules and their associated power amplifier boards will be described**

I. INTRODUCTION

In robotic systems, especially for humanoid robots, in the majority of systems a decentralized architecture is used [1],[2], [3], [4]. At the Research Center for Information Technologies (FZI) different kind of robots - like humanoid robots, four- or six-legged walking machines, mobile platforms and snakelike sewer inspection robots - are developed. For these robots we designed a computer architecture based on embedded PCs and distributed controller modules connected with each other via one or more CAN-Busses [5]. Though the requirements concerning the distributed components are quite different we wanted to implement a persistent design that could be used in all robots with only small amount of adaptation. In most robots built at our institute the main issues for the controller modules are space requirement, power consumption, several inputs for sensor value acquisition and communication interfaces (i.e. CAN-Bus). As none of the available of the shelf products suited all these needs we decided to build a controller module and an associated power amplifier ourselves.

II. COMPUTER ARCHITECTURE

The mechatronical construction of a robot can roughly be divided into mechanical aspects and in aspects of setting up the electronic and computer system. In this section we will describe how the electronic system of our robots is set up and how a computer architecture suiting the needs in these robotic systems was designed.

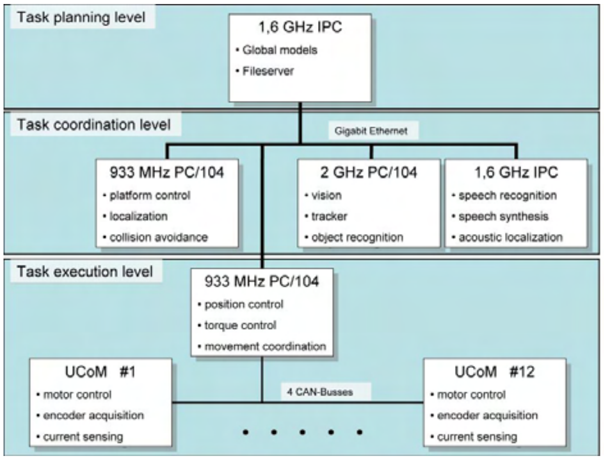


Fig. 1. Computer architecture used in ARMAR III

We started by identifying the concepts of how a robot should accomplish given tasks and thus proposing a control architecture that then was the basis for developing the computer architecture. We chose a hierarchically organized control system for the robots with the three following levels [6]:

- The task planning level specifies the subtasks for the multiple subsystems of the robot. Those could be derived from the task description autonomously or interactively by a human operator.
- The task coordination level generates in sequence/parallel primitive actions for the execution level in order to achieve the given task goal. The subtasks are established by the task planning level. The execution of the subtasks in an appropriate schedule can be modified/reorganized by an operator using an interactive user interface.
- The task execution level is characterized by control theory to execute specified sensory-motor control commands. This level uses task specific local models of the environment and objects, which represent the active scene. We call these models *active models*.

According to the control architecture the computer architecture is structured into three levels as well. Choosing suitable devices for these three levels yielded that the requirements of the task planning and task coordination level could be met with industrial PCs and PC/104 systems. As cabling in the robot is a major issue it is desirable to reduce cabling efforts as much as possible. Because of this we decided to use a

1-4244-0755-9/07/$20.00 ©2007 IEEE

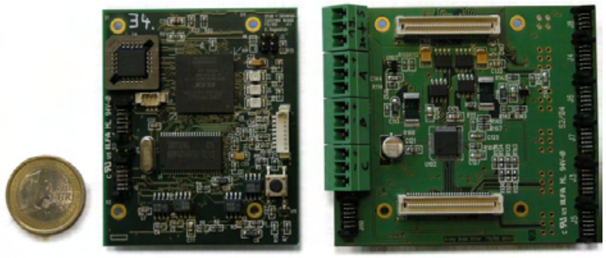
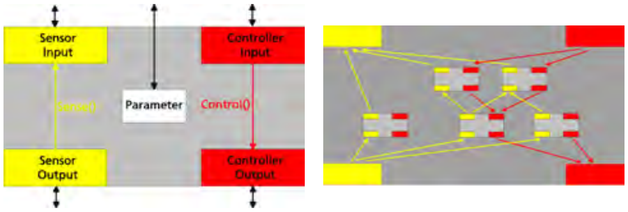


Fig. 2. The Universal Controller Module (UCoM) (left) and the 3way-brushdriver (right)

decentralized system for sensory-motor control. By placing controller modules close to the motors and sensors cabling can be reduced to one common power supply and a bus connection. Wires for supplying the motors and connecting the sensors to the controller can be kept short and have not to be passed through moving joints. To fulfil the requirements of the task execution level we designed the so called Universal Controller Module (UCoM) that in combination with our motor controller 3way-brushdriver (Fig. 2) is responsible for the sensory-motor control of the robot. The features of the combination of UCoM and 3way-brushdriver will be described in section III in more detail. For example in the humanoid robot ARMAR III (Fig. 5) these UCoMs build the basis of the proposed computer architecture in which we use a number of industrial PCs and PC/104 systems. These PCs are connected via switched gigabit ethernet. The connection to the lab PC is established by wireless lan on the master PC in the platform of the robot. To communicate between the UCoMs and the PC responsible for motion control we use four CAN-Busses to get realtime operation on the sensory-motor level. An overview over the structure of the computer architecture is given in Fig. 1. According to the controller architecture described above we use the following components

- Task planing level: One 1,6 GHz industrial PC. This PC establishs the connection to the lab PCs via wireless lan and acts as a fileserver for the other PCs on the robot. Furthermore it stores the global environment model
- Task coordination level: On this level we use one 933 MHz PC/104 system for the control of the platform including localization and collision avoidance, one 2 GHz PC/104 system running vision software like object recognition and a face tracker for human robot interaction and one 1,6 GHz industrial PC for speech recognition and speech synthesis as well as acoustic localization and separation of speakers and other sound sources. These PCs are responsible to gather sensor information like camera signals, laser scanner data, force torque values, audio signals etc. and distribute them to the task planning and task execution level
- Task execution level: On this level one 933 MHz PC/104 system and the UCoMs are used for position and torque control and coordination of movements. Depending on the task goal issued by the task planning level and the sensor values gathered by the task coordination level the sensory-motor control is accomplished



(a) One MCA module with the control- and sensor channels and the parameter input

(b) Illustration of dataflow between several MCA modules combined to a MCA group

Fig. 3. Dataflow in MCA module and MCA group

As software framework we use the Modular Controller Architecture (MCA2) [7] that was developed at our institute and is available under GPL online here [8]. The idea behind MCA2 is to structure the software into reusable modules (Fig. 3(a)) with simple interfaces. Each module has the three data channels: control data, sensor data and parameters. Via these data channels information is exchanged between the modules. A number of modules can be combined into a group (Fig. 3(b)) which has the same interfaces as one module.

III. CONTROLLER MODULES ON SENSORY-MOTOR LEVEL

Following the modular strategy that is realised in the software framework MCA2 we wanted to reach this modularity in the computer architecture as well. To achieve this goal not only for one of our robots but spanning all different robots we have to choose a common controller unit. This is important to reduce programming efforts as well. For example you can implement an PID-controller only once and as you use the same hardware that can run the same software in different robots you only have to adapt the PID parameters for the chosen joint. As already mentioned the main issues for the controller modules are space requirement, power consumption, several inputs for sensor value acquisition and communication interface. Especially for motor control the mandatory requirements for the controller module were

- Suitable to control three brushed DC motors at 24 V at up to 5 A
- Achieve cycle times as low as 1 ms
- Able to decode six quadrature coded signals (for each motor one from motor encoder and one from encoder at driven axis)
- Small outline, possible to be positioned close to the actuator
- Low power consumption
- Interface to access CAN-Bus

In some applications the 5 A might not be sufficient but this was the maximum current that could be realized without exceeding the space limitation. Besides electrical motors we use other actuation principles in our robots. For example in one of our walking machines - Airbug [9] - fluidic muscles were used and there is still ongoing research evaluating fluidic

TABLE I

DSP56F803 16-BIT HYBRID CONTROLLER

Feature	Characteristics
Speed	up to 40 MIPS at 80 MHz
Peripherals	CAN 2.0 B module, Serial Communication Interface, Serial Peripheral Interface, 16 bit external memory interface JTAG
Motor Control	6 PWM-channels, up to 2 Quad Timers
Package	100-pin LQFP
Memory	64 KB Program Flash, 1 KB Program RAM, 8 KB Data Flash, 4 KB Data RAM, 4 KB Boot Flash, up to 128 KB external program and data RAM
DSP features	16 x 16 parallel Multiplier-Accumulator (MAC), two 36 bit accumulators, 16 bit bidirectional barrel shifter

TABLE II

SPECIFICATIONS FOR UCOM

Component	Characteristics
Size:	57 mm * 70 mm * 15 mm
Weight	26 g
DSP:	80 MHz DSP56F803 (Freescale)
FPGA:	EPF10k30A (Altera)
Memory:	4k Bootflash, RAM, Flash, external RAM
Configuration-device:	EPC2-ROM
Peripherals:	CAN, SCI, JTAG, SPI
Motorcontrol:	6 PWM-Channels
AD-Converter:	2 * 4 Channels 12 bit AD-Converter
Programming:	Programable via JTAG or via CAN-Bus
Interface to piggyback board:	2 * 60-pin 0.8 mm pitch board-to-board connectors

muscles as actuation. For this kind of actuation a valve driver is needed. Furthermore in some robots like LAURON [10] we need extended sensor input like posture information from gyroscopes and acceleration sensors. To avoid building a special controller module for each of these applications we decided to split the controller module into one part that actually contains the controller and one part that contains the power amplifier, the valve driver or sensor acquisition electronics. As mentioned above we named the part with the actual controller Universal Controller Module (UCoM) as it will be universally used in our robots together with the respective piggyback board.

A. Universal Controller Module - UCoM

The choice for a suitable microcontroller/DSP for the UCoM could be narrowed down rather quickly as nearly no available controller had all the required features. The Freescale "DSP56F803 16-bit Hybrid Controller" (see Table I and [11]) came closest to our needs. This controller is a DSP featuring a set of peripherals usually only known from microcontrollers.

Though this hybrid controller nearly matches the requirements it still misses some essential features. On the one hand side it does not have enough general purpose IOs to control three motors on the other hand it only has two quadrature timers capable of decoding quadrature coded signals. To extend the DSP's flexibility we decided to put an FPGA next to it. As suitable FPGA we chose the Altera EPF10k30A. With this FPGA we can equip the UCoM with a high number of general purpose IOs. There are two great advantages to this combination of DSP and FPGA on the UCoM:

- We gain a high flexibility concerning routing and assignment of pins to the piggyback board. Through the FPGA we can reassign most of the signals so that it suits the used piggyback board
- We can use the FPGA to preprocess data that is exchanged between the UCoM and the piggyback board and further functionality can be implemented in the FPGA

By this approach we can disburden the DSP from tasks that are done in hardware more efficiently. For example we implemented six decoders for quadrature encoded signals. The communication between DSP and FPGA takes place via the external memory interface. As FPGA and the external RAM we integrated on the UCoM share the external address range we implemented an address decoder into the FPGA. This address decoder deasserts the chipselect for the lowest 64 addresses of the external address range and receives the sent data. For all other addresses the data is routed to the external RAM so that nearly the whole external memory range is available to the DSP and only the lowest 64 addresses are used as FPGA-registers. The DSP always initiates communication with the FPGA by writing to or reading from a FPGA-register. These registers are used to exchange data between the two devices. So to get the value of a quadrature coded signal all the DSP has to do is access the respective external RAM address.

The UCoM combines the Freescale "DSP56F803 16-bit Hybrid Controller" an external RAM and an Altera EPF10k30A on one board. For an overview of its features see Table II. It is interfaced to the desired piggyback board via two 60-pin 0.8 mm pitch board-to-board connectors. Via this connector the UCoM is supplied with a 5 V power supply. From this 5 V we generate the 3.3 V that are needed on the UCoM. We do not directly feed 3.3 V to the UCoM to avoid problems with voltage drop or disturbances that must be expected due to the wiring close to the motor power wires. The 3.3 V generated on the UCoM are then fed back to the board-to-board connector to be available on the piggyback board.

B. Motorcontrol Board

As the main actuation principle in our robots are electronic motors the first piggyback board we developed was a power amplifier able to drive three brushed DC motors. We named this piggyback board 3way-brushdriver. On the 3way-brushdriver we integrated three H-Bridges which are driven by a 3-phase brushless DC motor controller chip each. This motor controller chip can be configured to drive brushless or brushed motors. To drive brushless motors the hall-inputs to the driver must be connected to the hall-sensors of the motor. To drive brushed DC motors the hall-inputs are simply tied to ground. We chose this motor controller chip so that it can be interfaced by software in the same manner if we need to design a piggyback board for brushless DC motors in the future. In both branches of the H-Bridge we integrated a shunt

TABLE III

SPECIFICATIONS FOR 3WAY-BRUSHDRIVER

Component	**Characteristics**
Size:	80 mm * 70 mm * 20 mm
Weight	33 g
Power:	Three 24 V motors up to 5 A each
Encoder:	6 quadraturcoded signals or up to 24 General purpose IOs
Current sensing:	differential current sensing for each motor channel
Peripherals:	2 analog inputs, SPI, 3 General purpose IOs

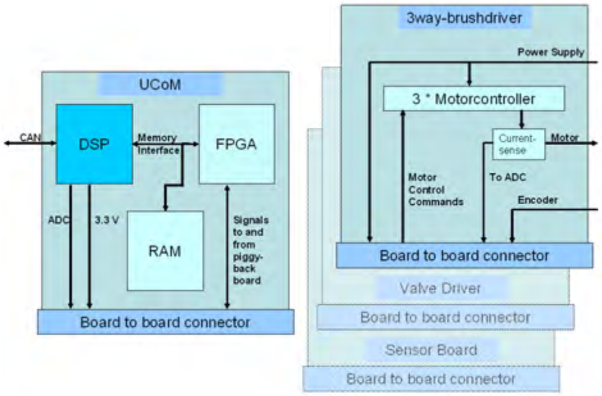


Fig. 4. Schematic overview of data flow on the UCoM and to the piggyback board

via which we can measure the motor current for each motor. As the motors are driven by PWM signals we had to use an OP-AMP with a high gain-bandwidth-product to amplify the signals for use in the AD-Converter.

As we wanted to keep the UCoM as small as possible we decided to put all interface connectors except the ones that are directly wired to the DSP like CAN-Bus, serial communication interface and JTAG to the piggyback board. Thus the piggyback board is responsible for supplying the UCoM with the 5 V input voltage. The 3way-brushdriver has a connector for input of the aforementioned 5 V, 24 V as power supply to the motors and a common Ground.

Further connectors on the motor control board are six small connectors for the quadrature coded encoder signals. Each of these connectors has six pins, two of which carry supply voltage of 5 V and ground for the encoder, two are the quadrature channels A and B. The remaining two are for encoders which have an index signal occuring each full revolution and for an pseudo absolute code strip we are evaluating. In addition to the encoder connectors the two remaining channels of the UCoM that are not used to measure the motor current are available on a four pin connector carrying a stabilized 3.3 V analogue supply voltage. An overview of the features of the 3way-brushdriver can be found in Table III, a schematic overview of the dataflow on the UCoM and to the piggyback board can be found in Fig. 4.

C. *Software Components on DSP*

To extend the modularity down to the code for the DSP each UCoM is treated as one module in the MCA2 software framework. Thus the UCoMs can be seamlessly integrated into the software running on the linux PC via a MCA-driver interfacing to the CAN-Bus. A number of different basic application programs for the UCoM were developed so that they can be used in most of the robots with only small adaptations. In the humanoid robot ARMAR-III (Fig. 5) we only use one generic program for all different joints. This generic program can be adapted via a configuration file that is evaluated at system startup. Other basic application programs that are implemented are:

- direct_pwm: A basic program for debugging purposes, essentially the PWM value for a given motor can be set and the encoder value can be read
- p_controller: A P-Controller that controls the motor via the respective encoder value. The P-Value can be set remotely
- pid_controller: A PID-Controller that controls the motor via the respective encoder value. The P-, I- and D-Value can be set remotely
- Speed/Position Controller: A cascaded speed and position controller

Further control programs including for example time discrete and torque control algorithms are in development.

To download these programs to the UCoM the JTAG-interface could be used, but if - in the development phase - code changes are quite frequently this is not suitable if the UCoM is not easily accessible in the robot. To avoid to plug in the JTAG-adapter to every UCoM each time the application code was changed we developed a bootloader for the UCoM. This bootloader resides in the bootflash of the DSP and was designed so that it is now possible to update the application code in the flash of the DSP via CAN-Bus. Thus it is necessary to program the bootloader only once via JTAG, after this all programming can be done remotely. At startup of the robot the UCoMs negotiate their position in the CAN-Bus automatically and get assigned a unique identifier via which they can be addressed. This is done with the help of an additional Init-Line which connects the Init-Output of one UCoM with the Init-Input of the following UCoM. The 11 Bit CAN-Identifier for addressing a specific UCoM is composed of the 5 Bit sequential number depending on the position of the UCoM in the CAN-Bus, a 1 Bit type identifier and the upper 5 Bit describe the meaning of the following data bytes. After this initialization phase newer versions of the application on the DSP are automatically downloaded and then executed.

D. *Functions on FPGA*

The FPGA can be seen as an extension to the DSP. It equips the DSP with a large number of general purpose IOs and processes data that is exchanged between the DSP and the piggyback board. As stated above we use the memory interface to communicate between DSP and FPGA. The function blocks in the FPGA are programmed in VHDL. There are some blocks that are common to all designs independent of the piggyback board:

- Address Decoder: This block evaluates the address and decides wether the data is directed to the FPGA or the external RAM. There are 64 read-write-register implemented for data exchange between DSP and FPGA
- Version supervision: We use four different version strings to identify which program is running on the FPGA
  - Board-ID: This is a four-bit ID checking if the piggyback board and the FPGA program match
  - Application-ID: One piggyback board can be used in different scenarios requiring a different FPGA program. This is mirrored in the Application ID
  - Major and Minor version: These are numbers used for revision management
- Initialization: This block is responsible for assigning pin levels while negotiating the position in the CAN-Bus. During Initialization the LEDs on the UCoM mirror the state of the initialization pins, during normal operation they show the state of the UCoM
- Watchdog: This unit checks wether the DSP is still active and returns to a safe state by disabling the motor outputs of the piggyback board if the DSP is not operating properly

For the introduced 3way-brushdriver the following function blocks were already implemented into the FPGA:

- Quadrature decoder: This unit decodes the quadrature coded signals coming from the encoder inputs. It can be reset when an index event occurs and features a prescaler so that no counter overflows are encountered
- Pseudo Absolute Decoder: Evaluation of additional pseudo absolute track on our encoder strips
- Serial Synchronous Interface: A protocol for interfacing with various sensor types like magnetic or optical absolute encoders
- Motor Control register: This block transfers the value of the motor commands - like direction, enable or break - to the motor controller

Further modules can be integrated into the FPGA as they are needed.

IV. ROBOTIC PROJECTS

The proposed concept was successfully implemented in some of our current robots. For example in the humanoid robot ARMAR III (Fig. 5) the computer architecture is set up as depicted in Fig. 1. In total five embedded PCs and 12 UCoMs are employed. Due to the high number of controlled axes there are four CAN-Busses in use. In ARMAR III we managed to reduce the number of application programs for the UCoMs to one generic program that can be configured remotely.

Fig. 5. The humanoid robot ARMAR III

In the six-legged walking machine LAURON IV the same concepts are realized (Fig. 6). In LAURON IV one PC/104 system and seven UCoM fulfil the control tasks. In this walking machine the six UCoMs controlling the legs share one controller program, only the controller for the head had to be slightly adapted.

In the snake-like sewer inspection robot MakroPlus the space limitations were much stricter then in the other robots, so that a dedicated board (Fig. 8) had to be designed . Though a new board had to be produced it could be realized as a combination of UCoM and 3way-brushdriver on one board. Thus it fits into the modular design in the same way as the UCoM itself and the same basic software can be used.

Though the above mentioned robots are quite different they share the common concept of a modular software framework with its respective computer architecture and realization in electronic hardware.

V. SUMMARY AND OUTLOOK

In this paper we presented a modular concept to control robots. This concept includes a control concept from which the used computer architecture was derived, a modular software framework and the development of a hardware architecture that can be mapped into the computer architecture. The focus of this paper was on setting up a modular system that can be used in a variety of robots so that not only software components can be reused but that also the hardware is interchangeable. It was laid out how this goal was achieved and especially the development of the UCoM and the 3way-brushdriver were described. Some examples of our robots in which these control

Fig. 6. Six-legged walking machine LAURON IV

Fig. 7. Snake-like sewer inspection robot MakroPlus

devices are successfully used were presented in the above section. In ongoing research we will implement this concept in robots that are going to be built. Most likely further piggyback boards will be designed for that purpose.

VI. ACKNOWLEDGEMENT

This work is partially supported by the German Research Foundation (DFG) within the Collaborative Research Centre SFB 588 (Humanoid Robots - Learning and Cooperating Multimodal Robots).

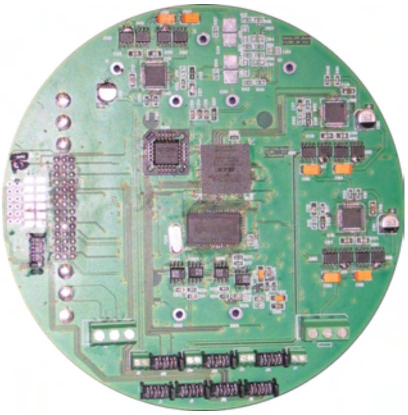


Fig. 8. Dedicated controller for snakelike robot - Combined UCoM and 3way-brushdriver

REFERENCES

[1] J.-Y. Kim, I.-W. Park, J. Lee, M.-S. Kim, B. kyu Cho, and J.-H. Oh, “System Design and Dynamic Walking of Humanoid Robot KHR-2,” in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, 18-22 April 2005, pp. 1431–1436.

[2] T. Sugihara, K. Yamamoto, and Y. Nakamura, “Architectural design of miniature anthropomorphic robots towards high-mobility,” in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2-6 Aug. 2005, pp. 2869–2874.

[3] J. Butterfass, G. Hirzinger, S. Knoch, and H. Liu, “DLR’s multisensory articulated hand. I. Hard- and software architecture,” in *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 3, 16-20 May 1998, pp. 2081–2086vol.3.

[4] R. Bischoff and V. Graefe, “HERMES - a versatile personal robotic assistant,” *Proceedings of the IEEE*, vol. 92, no. 11, pp. 1759–1779, Nov. 2004.

[5] K. Regenstein and R. Dillmann, “Design of an open hardware architecture for the humanoid robot ARMAR,” in *Humanoids 2003, International Conference on Humanoid Robots, Conference Documentation, October 1 - 3, 2003, Karlsruhe and Munich, Germany*, October 2003, p. 3. [Online]. Available: http://www.fzi.de/KCMS/kcms_file.php?action=link&id=220

[6] T. Asfour, D. Ly, K. Regenstein, and R. Dillmann, “Coordinated Task Execution for Humanoid Robots,” in *Proceedings of ISER 2004, The 9th International Symposium of Experimental Robotics*, 2004. [Online]. Available: http://www.fzi.de/KCMS/kcms_file.php?action=link&id=420

[7] K. U. Scholl, J. Albiez, and B. Gassmann, “MCA - An Expandable Modular Controller Architecture,” in *3rd Real-Time Linux Workshop, 2001, Milano, Italy*, 2001. [Online]. Available: http://www.fzi.de/KCMS/kcms_file.php?action=link&id=536

[8] FZI, “Modular Controller Architecture Version 2.” [Online]. Available: http://www.mca2.org/

[9] K. Berns, J. Albiez, V. Kepplin, and C. Hillenbrand, “Airbug - Insect-like MachineActuated By Fluidic Muscle,” in *Proceedings of CLAWAR 2001, Int. Conference on Climbing and Walking Robots*, 2001. [Online]. Available: http://www.fzi.de/ids/publikationen.php?id=413

[10] B. Gassmann, T. Bär, J. Zöllner, and R. Dillmann, “Navigation of Walking Robots: Adaptation to the Terrain,” in *Proceedings of CLAWAR 2006, 9th International Conference on Climbing and Walking Robots (CLAWAR)*, 2006.

[11] Freescale, *DSP56F803 Data Sheet*. [Online]. Available: http://www.freescale.com/files/dsp/doc/data_sheet/DSP56F803.pdf