FreeScale Microcomputers Automation and Process Control

VLADIMIR VASEK, PETR DOSTALEK, JAN DOLINAY
DAGMAR JANACOVA, KAREL KOLOMAZNIK
Tomas Bata University in Zlin
Faculty of Applied Informatics
Mostni5139, 760 01 Zlin
CZECH REPUBLIC
, www.fai.utb.cz

Abstract: - This contribution deals with employing 8 and 16-bit Freescale microcomputers in process control with a view to implementation standard and modern control methods such as adaptive and robust controllers. Although microcomputers have limited amount of memory and computing power it is possible to successfully use them in this area and bring us improved regulation quality with cost saving. The contribution outlines the programming technique used to program the 8-bit microcomputers Freescale 68HC11and 68HC08 and the 16-bit microcomputers 68HC12 in assembler and library of program modules for monitoring and control applications. Further it describes the possibilities of programs transfer between mentioned types of controllers.

Key-Words: - Microcomputers, 68HC11, 68HC08, 68HC12, Program Modules, Software Converter.

1 Introduction

There is now wide choice of computer systems which can be utilized for process control and automation of the technological processes ranging from powerful industrial PCs to cheaper microcomputers and programmable logic controllers. The new generation of 8-bit and 16-bit microcomputers is so powerful, that it is possible to use them for most industrial applications of the monitoring and control systems. Possibilities of including them to the distributed systems increase their signification to solve partial control loops this way. We can use microcomputers for classical discrete control loops by the help of standard discrete control algorithms however, on the basis of their features it is not possible to exclude them from the applications of the complicated and time-consuming control algorithms. In this contribution there is described using of the user library focused to the monitoring and control which is written for the three types of Freescale microcomputers - 68HC11, 68HC08 and 68HC12. 68HC11 is representative of older terminate generation and most of library parts were written for them. That is why were developed possibilities to transfer of the program modules to members of the new microcomputers generation. This work presents our experience with implementing standard and some modern control methods on Motorola HC11 microcontroller and possibilities of further progress with new powerful microcontrollers HC08 and HC12, whereas simple changeover to new hardware is possible through developed software converter.

2 Program library for monitoring and process control

Higher programming languages are generally preferred today but sometimes it is useful or necessary to program in assembler, especially with microcontrollers. The reason may be that the maximal speed of the program is required, the memory is small or there is no good higher-level language compiler available for the microprocessor. However, undeniable disadvantage of assembler when compared to higher programming languages besides the troublesome portability of programs is also low productivity of labor. One of the ways of making assembler programming more efficient is to use a library of pre-created modules or subroutines. With the help of this library developer should be able to take required modules and put them together to form a new application with minimal changes and little new code written. Currently our program library contains following basic modules: digital input/output, analog input, three-state controller without and with the penalty, PSD controller (including Takahashi's modification) and general discrete linear controller.

From the area of the modern control methods there are modules for recursive identification, robust controller, and two self-tuning controllers based on required model method and pole placement. The modules are provided as .ASM files, which contain not only the code of the module but also definitions of required constants and masks.

2.1 Basic modules

Library contains modules for handling digital input/output ports with possibility to mask unused inputs, negate selected inputs/outputs and others. A/D input module can process up to eight channels of analog input signals using A/D converter integrated on the microcontroller. Next part of library contains modules of four types of controllers: three-state controller without and with penalization, PSD controller, Takahashi's modification of PSD controller and general linear discrete controller.

2.2 Modules for modern control methods

Modules dedicated to modern control methods include module for recursive identification based on least squares, which allows to calculate controlled plant parameters in real time. Next there are two modules for controller parameter synthesis – one, which makes use of required model method and another one, which utilizes the pole placement method. Besides that a robust controller module was also developed. This module can be used for tuning of robust controller parameters.

2.2.1 Adaptive control

Majority of real processes has stochastic character and their parameters can be considered constant only with higher or less degree of incorrectness. Many factors can induce change of the parameters of a process, for example change in the operating mode, different quality of the fuel or raw materials or change of the properties of the plant itself, caused for instance by aging.

Adaptive system can be defined as a system, which measures behavior indexes of the plant, compares them with desired indexes and modifies parameters or structure of the system or generates auxiliary signal so that the measured indexes get near the desired indexes.[2] The behavior index can be for example zeroes or poles of the transfer function, overshoot of the step function, the time of regulation, minimal values of various integral criterions or frequency spectrum.

The classification of adaptive systems is not unified yet, one of the possibilities is:

- Heuristic adaptive controllers
- Model reference adaptive systems (MRAS)
- Systems with variable structure
- Self tuning controllers (STC)

For our program library was used method of self tuning controllers. These controllers are based on continuous estimation of controlled plant characteristics and their gradual refinement. Based on this knowledge an optimal controller is then proposed. This procedure makes it possible to catch changes in the controlled plant parameters, improve the regulation process when disturbances are present and also enables to automatically set up controller parameters.

2.2.2 Problems of Identification for adaptive control

The conditions for identification in the process of adaptive control are not ideal. It is necessary to keep in mind following presumptions:

- Data (input) is generated by the controller
- The aim of the controller is to eliminate disturbances and stabilize the process. That makes conditions for identification more difficult.
- The process of identification for adaptive control takes long (infinite) time. It is hardly acceptable to assume that the estimated parameters are constant and therefore methods of estimating time variable parameters must be used.
- The identification must work in different modes of the plant – in virtually stationary state, when disturbances are present or during shifting between various states.
- The structure (order) of the model usually cannot be changed during the process.
- The algorithm must be quick and reliable.

2.2.3 Controller synthesis

Basic part of an self tuning controller is the block that computes parameters of the control law (parameters of the controller). Presently PID controllers are mostly used in the industry. Use of these controllers has a long tradition, there are many methods available for tuning these controllers and the suppliers as well as users have wide experience with them. All these circumstances contribute to the fact that in real-world applications simple adaptive controllers are more common than adaptive controllers based on theory of optimal control. Generally it is possible to use the same methods for self-tuning controller as for synthesis of conventional controller. The only limiting factor can be the computing power demanded for the algorithm. That is because during each sample period not only the parameters of the controllers must be computed, but also one step of identification of the controlled plant together with other operations must be performed. It is therefore necessary to choose such an algorithm that will be feasible on a given microcomputer with given sample period.

For the library of program modules two well tried and in conventional controllers widely used methods were chosen – required model method and the pole placement.

The required model method

This method was developed for tuning of conventional controllers [7]. It allows tuning discrete or analog controller so that defined overshoot is achieved for stepwise reference or disturbance on the output of the plant. Compared to well-known Ziegler-Nichols method this method is more accurate and universal while the same simplicity is preserved. The transfer function G_R of a controller, which will ensure desired transfer function G_w of the feedback system

$$G_w = \frac{Y}{W} \qquad (1)$$

Is given as follows

$$G_R = \frac{G_w}{G_s(1-G_w)} \qquad (2)$$

This equation is based on a very general method of required model method, which is also known as compensating method.
We suppose system with discrete controller, where the transfer function is as follows

$$G_w(z) = \frac{k_{od}T}{z-1+k_{od}Tz^{-d}}z^{-d} \qquad (3)$$

$$d = \frac{T_d}{T} \qquad (4)$$

k_{od} is the gain of open-loop system with discrete controller, T – sample period, d – discrete tranfer delay.
Using the procedure given in [7] we will obtain the transfer function of the controller:

$$G_R(z) = \frac{aT}{(z-1)G_S(z)}z^{-d} \qquad (5)$$

Pole placement

Controller based on the placement of the poles of a feedback system is designed so that it stabilizes closed feedback loop whereas the characteristic polynomial has pre-defined poles.

Besides the stability it is quite easy to achieve required course of the output signal (for example maximal overshoot, damping etc.) [1]

For FB system the synthesis consists in solving the diophantine equation

$$AFP + BQ = D \qquad (6)$$

Where F is the denominator of the transfer function of reference signal, $\frac{Q}{FP}$ is the transfer function of the controller and $\frac{B}{A}$ is the transfer function of the plant to be identified. D is system characteristic polynomial:

$$D(z^{-1}) = 1+\sum_{i=1}^{n_d} d_i z^{-i}\ ,\ n_d \le 4 \qquad (7)$$

2.2.3 Robust control

The problem of robust control can be summarized as follows: We have a controller with transfer function

$$G_R(s) = \frac{q(s)}{p(s)} \qquad (8)$$

which was designed for a plant with transfer function

$$G_S(s) = \frac{b(s)}{a(s)} \qquad (9)$$

This plant is called nominal plant. In the real world the controller controls a plant

$$G_S'(s) = \frac{b'(s)}{a'(s)} \qquad (10)$$

which more or less differs from the nominal plant. This second plant is called perturbed plant. The problem is to determine whether the controller designed for the nominal plant will be good also for the perturbed plant, i.e. whether the controller is robust or what conditions must be satisfied if the controller should be robust. Robustness of a controller thus means that the system preserves certain qualities not only with the transfer function it was designed for, but also for certain neighborhood of perturbed plants. Elegant way of design and tuning of robust controllers is provided by algebraic theory. The principle is quite simple: By solving the diophantine equation all stabilizing controllers are found and finite controller is then chosen according to divisibility conditions. This way controller can be easily parameterized with a single number m>0. The only problem is that this procedure cannot be performed in the ring of polynomials but it is necessary to convert to the ring of stable rational functions R_{ps} [3].

2.3 HC11/HC08/HC12 Conversion

Due to the compatibility of the instruction file of the families of single-chip microcomputers FreeScale HC11 and HC12, we were looking for a reduction of the software portability from HC11 to HC12.

In the case of HC08 family using is the principle of the transformation defined by help of the special software.

For solution HC11/HC12 compatibility we chose a specific representative of the HC12 family – MC9S12E128CPV for hardware and as for software, we chose a user library of modules for monitoring and control of technological processes developed for the HC11 microcomputers. A part of this library is also the standard library for working in floating point and a real-time operation system, which is necessary for the above-mentioned kind of applications. [8]

2.3.1 HC11/HC12 Conversion

To transfer the programs to the HC12 microcomputer, it is necessary to formally adapt the source texts due to different notation syntax in HC11-as11 translator and the translator for HC12 integrated within the development environment CodeWarior. For this adaptation, the "HC11toHC12" program has been compiled with the use of the Visual Basic language. This work presents implementation of the program modules on a member of wide family of 8-bit Motorola's HC08 microcontrollers. Concretely general-purpose microcontroller 68HC908GP32 was chosen which is suitable for wide range of applications.

Generally there is a principle that the adaptation of source modules via this converting program runs trouble-free in cases when the programs do not turn right on the hardware components of the microcomputer. If so, it is necessary to distil the applied hardware components into the program equipment; in some cases only to modify their addressing.

2.3.2 HC11/HC08 Conversion

For this conversion was built the special conversion program. The aim of this software tool for program conversion is to transform assembly language source files of M68HC11 in the format of compiler AS11 to the source files, which can be assembled using the compiler CASM08Z and subsequently executed on M68HC08 microcontrollers with the same results. Regarding to smaller number of M68HC08 central processing unit internal registers, all have been replaced by variables located in the first page of internal RAM at address space $40-$FF. These memory locations are called "virtual registers", because they are supplying all registers from M68HC11 microcontroller. Nonexistent instructions, especially arithmetic and logic with 16-bit operands, are emulated with more basic instructions of M68HC08 microcontroller resulting in slower execution speed in some cases. Next compatibility problem is hidden in condition code registers because positions of the each flag are not identical (only C flag is on the same position). This issue with swapped flags in CCR affects instructions TAP and TPA that transfers content from accumulator A to CCR and vice versa. It is solved by swapping flags in CCR register when these instructions are called. Branch operations with relative addressing mode, such as conditional branches, are extended to absolute addressing mode, because converted source file is always longer, so jumps would be out of range. Individual instructions of M68HC11 are converted to the macros or subroutines (used method depends on optimization type setting), which completely substitute their function on M68HC08. Instruction replacements are stored in two separate files - the first one contains macros substituting all M68HC11 instructions. The second file contains subroutines, which can be used only on subset of whole instruction set.

3. Conclusion

To support the implementation of standard and modern control methods on the Freescale 68HC11, 68HC08 and 68HC12 microcomputers the following modules were developed as a part of program library module: digital input/output, analog input, three-state controller without and with the penalty, PSD controller (including Takahashi's modification) and general discrete linear controller, next for recursive identification, which employs the least squares algorithm with adaptive forgetting and modules for controller synthesis based on the required model method and on the pole placement methods. Besides that a robust controller was also implemented. To verify the functionality of the modules several applications were assembled, for example application for continuous process identification, adaptive controller based on required model method and other. These applications were then used for controlling of a real system. It proved that it is possible to employ modern methods of automatic control on existing hardware and achieve the benefits these methods offer, such as simple utilization, better quality of the control or energy savings.

Acknowledgement

This work has been supported by the Ministry of Education of the Czech Republic in the range of the project No. MSM 7088352102.

References:

[1] Doyle, J.C., Francis, B.A., Tannenbau, A.R.: Feedback Control Theory, Maxwell Macmillau Company, New York, 1992

[2] Vidyasagar, M.: Control System Synthesis, The MIT Press, Cambridge, Massachusetts, London, 1987

[3] Prokop, R., Corruou, J.D.: Design and Analysis of Simple Robust Controllers. Int. J. Control, 1997, vol. 66, č. 6, 905-921

[4] Motorola 68HC11 Reference manual.

[5] Motorola 68HC08 Reference manual.

[6] Motorola 68HC12 Reference manual.

[7] Viteckova, M. : Seřízení regulátorů metodou inverze dynamiky. Skriptum VŠB – Technická Univerzita Ostrava, 2000.

[8] Vasek, V., Dostalek, P., Zeravik, M., Janacova, D.: Software Compability FreeScale Microcontrollers HC08, HC11 and HC12. Wseas Trans. On Computers. 12, Vol.5, s.3000, 2006, ISSN 1109-2750