Registration Number: 4EQ2

SAM - <u>S</u>emi <u>A</u>utonomous <u>M</u>icro Air Vehicle

Jens Altenburg

1 Abstract

Autonomous operating systems, often called robots, fascinate with their behaviour. Most of those systems are ground based, that means they have wheel or legs for moving. Robots cannot fly, can't they?
There are some experiments existing like the so called UAV (unmanned air vehicle). Most of them have been developed by universities and sponsored by huge companies.
Now, in case of no need of complete autonomous flight an interesting flying vehicle is presented. The combination of a fabulous swept-forward tailless aircraft model and the very fast SX microcontroller made it possible to design a semi autonomous micro air vehicle, called SAM.
The special needs of the swept-forward tailless design can be solved by an on board computing system. Semi autonomous means, that a pilot flies the aircarft with a radio, but some functions are supported by electronics.
But with all the enthusiasm, the aircraft has to be built as cheap as possible, so that not only rich people can fly, but also students and pupils. The schematics and the basic control software will be published, so everyone has a chance to understand the design and/or improve it.

2 Idea

There are a huge amount of radio controlled aircraft models available. You can buy cheap foam made slow- and funflyers, or you buy or build expensive semi-scale RC models. If you are a beginner in flying RC models, the lifetime of your model is calculated in minutes. To be a good pilot is a hard job.
Now, there is a new material available, the so called EPP. The material looks like normal styro, but is nearly indestroyable, really!
I got a new developed EPP model, the "Bat" for testing (manufacturer: *www.acteurope.de*). The model designed by Klaus Westerteicher from ACT-Europe (*www.acteurope.de*) is a so called swept-forward tailles aircraft (SF). OK, it looks a little bit strange, but it has an wonderful flying behaviour. There is an excellent describtion about SF aircrafts in [2]. We need to know that a well designed aircraft cannot be stalled.
Another advantage is the easy to built assembly kit. Only a few componets have to be mounted. And last but not least the model is cheap.
With such an amount of advantages, there are also some disadvantages. Because the aircraft is controlled by only two ailerons both servo signals have to be mixed. If you are an owner of a computer radio, you program your radio with the mixing parameters. Especially this point looks like a contradiction; cheap model - expensive radio. That's not a good combination for beginners.

Another disadvantage is the roll control of the model during approach. Because thereare only two ailerons existing, they have to be used as an aileron/elevator combination. During approach (low altitudes) only small swings are allowed.

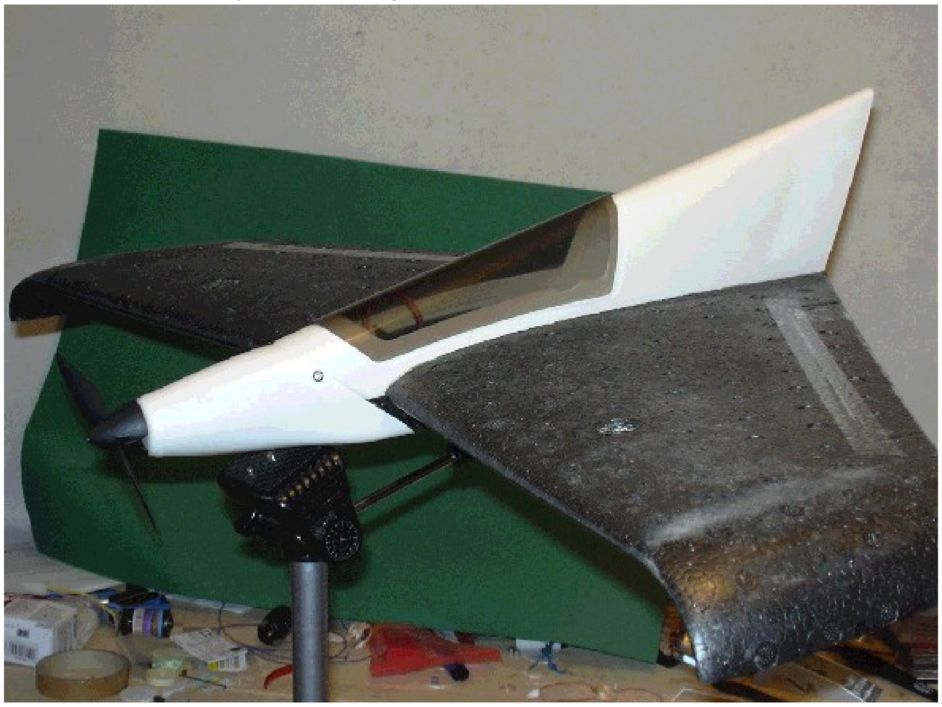


Figure 1 Swept-forward tailles aircaft model

Now it's time for the SX based flight control computer (FCC). Let us define some base lines for such an equipment.

- Using of standard 3 channel radio
- Internal mixing capability for two servos
- Third channel controls the engine (PWM signal is generating by the FCC no external motor controller is needed)
- Failsafe function (detecting male function of receiver, noise, overrange, etc.)
- Stabilisation of the aircarft in roll axis
- Option 1: altitude depending aileron swing
- Option 2: second radio link (ISM-Band) for data transmission, like servo positions, height, temperature, speed, etc.

Option 1 and 2 are prepared on the pcb. Because of the limited RAM size of the SX28 (and the limited ROM code size of the CC1B, abrox. only 1550 words) both options are not included in the basic software system.

3 Hardware design

3.1 Core

The SX28AC microcontroller is the core of the flight computer, figure 1. A lot of interfaces exist, JP5 connects the RC receiver to the micro, JP1...JP4 are the output connectors for servos and engine and JP6 is the external interface (SPI - serial pheripheral interface). U8 is the air pressure sensor, U1 the optional radio link (ISM band transceiver). Figure 2 shows the schematic of the attitude detector. For programming/debugging JP7 is used.
The SX chip runs with a clock speed of 50 MHz at a voltage of 3.3 V.

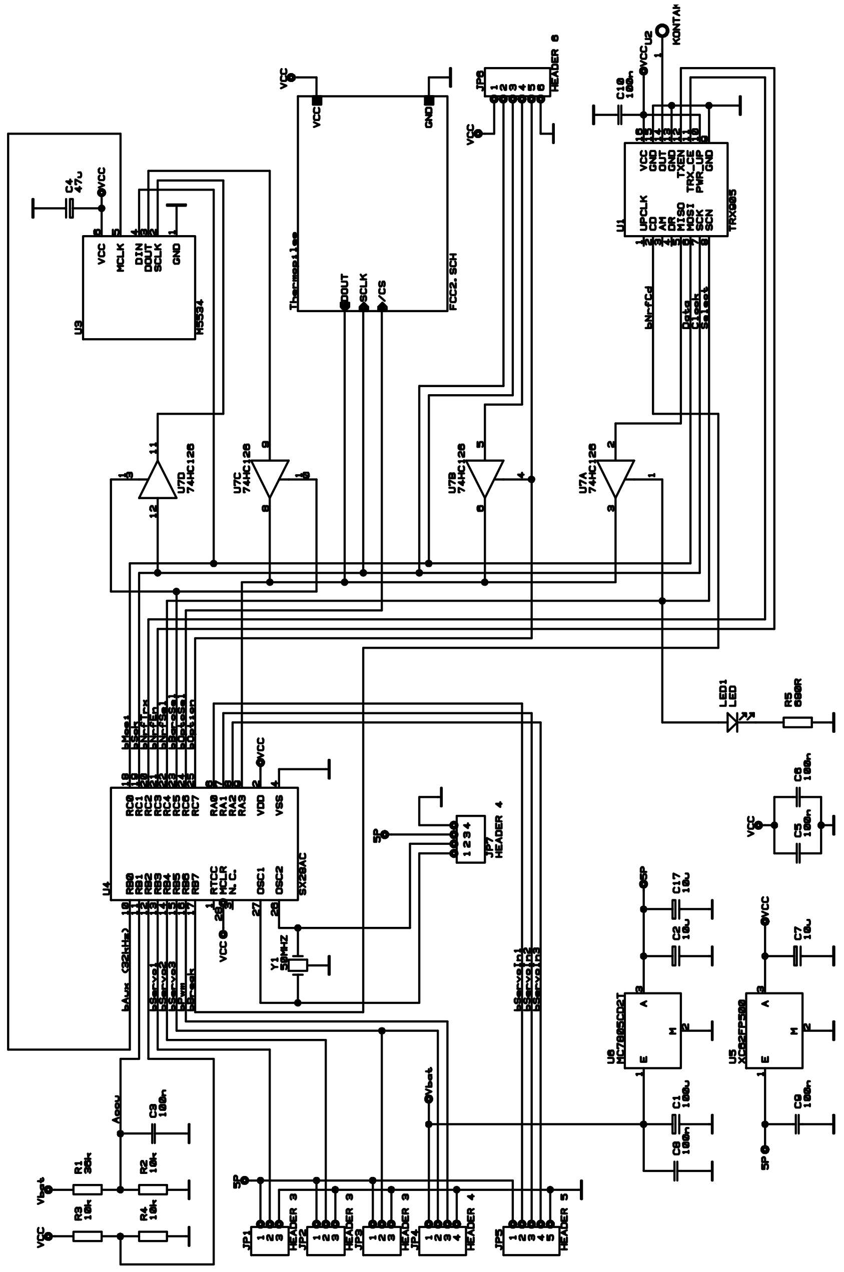


Figure 2 Schematic of the core of the flight computer

3.2 Power supply

There are two voltage regulators on board 5 V/1 A and 3.3 V/200 mA. The input voltage has to be in the range of 7.5 ... 12 V. JP4 is the power input connector. Because of the strong weight limitations of the model aircraft using a lithium polymer accu is highly recommended.
The regulator U6 is needed for powering the servos and the RC receiver. U5 powers the microcontroller and the 3.3 V circuits. For minimizing power losses the input of U5 is driven by the 5 V output of U6 instead of connecting to the accu.
Overcharging accus has to be avoided strictly. R1...R4 and C3 are the external components of the comperator circuit of the microcontroller. R3 and R4 defines the reference level (Vcc/2 = 1.65 V). The recomended lowest accu voltage is about 7.5 V (3 cell lipo accu with 11.1 V nominal voltage). R1 and R2 shift the accu voltage in the range of reference voltage. C3 decreases noise. If other accus are wanted to be used, the R1 and R2 values have to be changed.
A software routine checks the status of the comperator output and disconnects the engine (dc motor) from the accu; the PWM signal goes low and the break signal goes high. Note, that disconnecting the engine decreases accu load, but the flight control computer is active as well. The aircraft has to be landed as fast as possible.

3.3 ISM-Band radio link

In some cases it would be very helpful if some data from the aircraft are available in realtime. The used ISM band module is small enough to be integrated into the flight

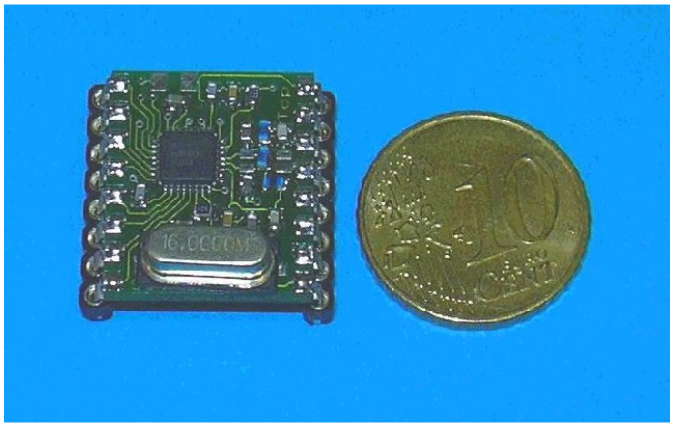


Figure 3 ISM band module

computer.The module has an output power of 10 mW and a frequency of 815 MHz (Europe) or 915 MHz (USA). The maximum data rate is 50 kBaud (50,000 bit per second). A software module for the ISM band module is available as well as ground base station hardware.

3.4 Air pressure sensor

The other optional component is the air pressure sensor M5534 from Intersema Sensoric SA (*www.intersema.com*). This component consists of the pressure sensor itself and some extra

logic for calibration. The resolution of the sensor is about +/-1.5 mbar, that's enough for an altimeter resolution of about +/- 1 m (approx. 3 feet).
The sensor is connected via a simple serial interface. A software module for reading the circuit can be given, but a larger SX chip has to be used because of RAM limitation.

3.5 Horizon detection unit

After all the optional (also non-existing features) the horizon detecting unit is one of the special features of the flight control computer.
To avoid fatal crashes some years ago a so called autopilote sytem was sold. The HAL 2100 autopilote measures the light differences between air and ground and stabilized the attitude of the airplane.
The principle of the optical autopilote was very simple. Four LDRs (light dependent resistor) are placed, so that light intensity in four different directions (front, back, left and right) can be measured. A microcontroller compares the measured intensities and calculates control commands for the servos if necessary.
Under normal light conditions the systems works, but direct sunlight blinds the LDRs, so that the autopilote fails. Another problem was the correct mounting of the sensor case. Small differences between normal flight leveling and sensor horizon leveling produces control command errors and destabilises the aircraft.
To avoid the effects through direct sunlight in [1] an interesting application is shown which uses special sensors.

3.5.1 Theorie of Operation

As descibed in [1] the so called thermopile sensors measure temperature differences instead of different light intensities. The main idea is the measurement of temperature differences between sky and ground. Typical sky temperatures (sensor in zenith direction) are in the range of -10...-15 °C (258...263 Kelvin) while ground temperature values about 10 °C are measured. These differences should be used to detect bank angles of an aircraft vehicle.
Before we discuss the horizon detection sensor, let us have a look at the temperature sensor, the so called Thermopile.
A thermopile is a serially interconnected array of thermocouples, each of them consisting of two dissimilar materials with a large thermoelectric power (voltage) and opposite polarities. The thermocouples are placed across the hot and cold regions of the structure and the hot junctions are thermally isolated from the cold junctions. The cold junctions are typically placed on the silicon substrate to provide effective heat sink.
The black body in the hot region absorbs the infrared rays. In the result a temperature difference produces a thermoelectric voltage. The sensor responds to a broad infrared spectrum, aboud 5...13 µm wavelength, does not require a source of bias voltage or current. The measured voltage depends on the difference between object and sensor body temperature.

But why isn't the sensor blinded by direct sunlight, or why does the distance of the measured object doesn't have any effect of the output voltage? Well, with (very) easy words the thermopile works in comparision to the colour detector. Different object temperatures relate to different colours. For example, if you see a red object, like a traffic sign, the object is red independly how far away it is. The light intesity goes down in large distances, but the colour won’t change. That's it.

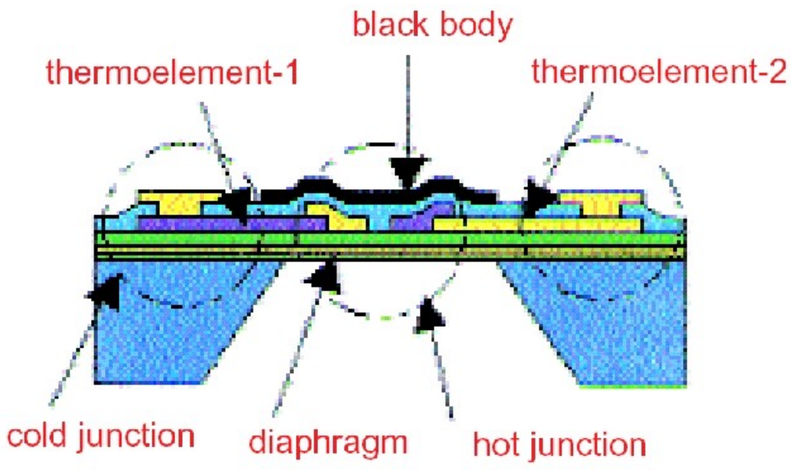


Figure 4 Cross section of the thermopile

The other point, direct sunlight doesn't have an effect, because the sensor works without a lens, etc. The wide angle of view integrates the temperatures of all objects in the viewed area. An integrated silicon filter supports this behaviour. Even if the sensor views directly into the sun, the area of the hot point, the sun, is very small in comparision to the rest of the viewed area.

3.5.2 Horizon sensing

The easiest way for horizon sensing is the use of two thermopile sensors. The sensors are located in opposition, one views right, the other views left.

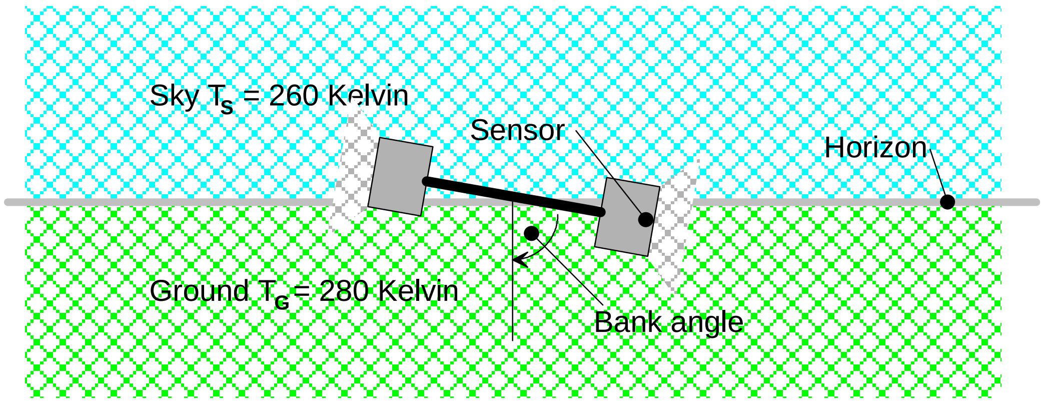


Figure 5 Horizon sensing through temperature differnces

Picture 5 shows typical measured values. In level flight both sensors "view" the same parts of sky and ground temperature. The output voltage of both thermopiles is equal. In the picture the right sensor views the ground more than the sky, and the left sensor works the other way round. The resulting temperature of the right sensor is larger than the left. To correct the flight attitude steering commands have to be given as long as the values are not equal.

3.5.3. Schematic of the horizon sensor

Figure 6 shows the schematic of the horizon detector. Two serial connected thermopiles view the opposite side of the aircraft. The difference voltage is amplified by an OPV.

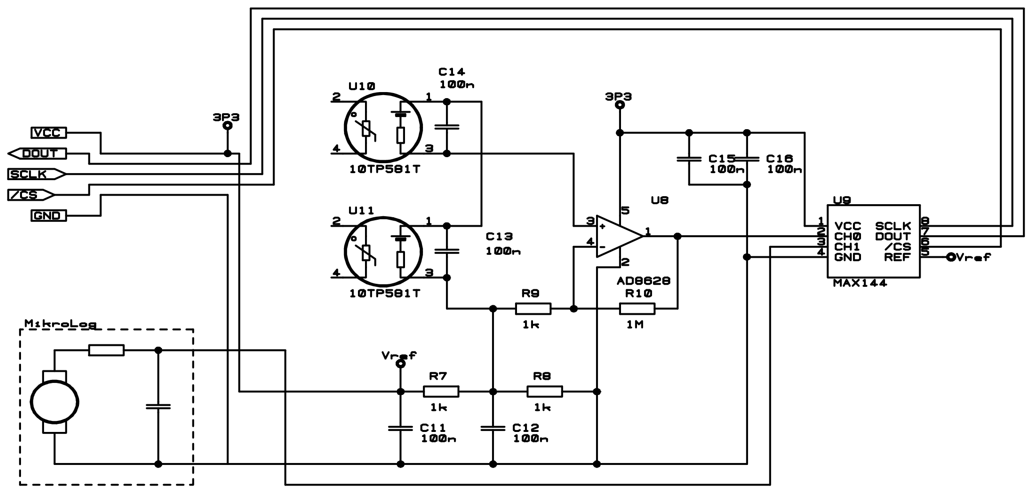


Figure 6 Horizon detector

As long as both detectors view the same angle of sky and ground, the difference voltage is nearly 0V. The output voltage of the OPV is $V_{out} = V_{ref} + V_{diff} * (R10/R9)$. Channel 1 of the A/D converter reads this value.

Note, the output voltages of the thermopiles are in the range of a few millivolt, difference voltages are in the range of microvolt. That means the used OPV has an offset voltage as low as possible. The AD8628 from Analog Devices (*www.analog.com*) has a typical offset voltage of 1 µV.

Channel 2 measures the voltage which depends on the revolution of a miniature motor. The motor is driven by a small propeller and the output voltage depends on the revolution speed. This simple "microlog" measures the airspeed of the model aircraft. The basic software doesn't use this value.

4 Softwaredesign

4.1 Block structure of the flight control computer

In the adaption on our semi automoumus aircarft it has to be expected, that we need only three independ control channels, aileron/elevon and engine.

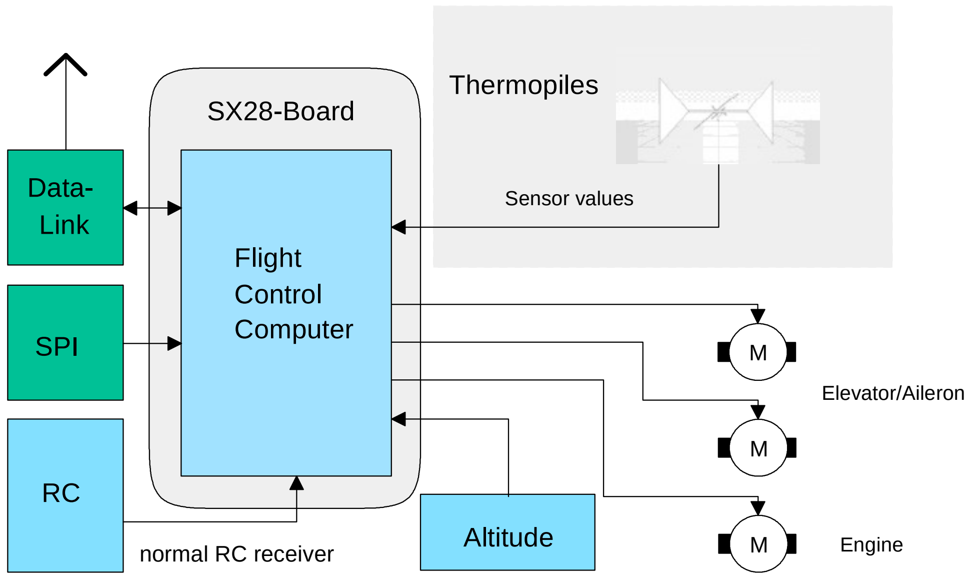


Figure 7 System design of the complete flight control computer

Figure 7 gives an overview about the different connections of the SX board to the several sensors or servos. The minimum configuration includes RC (radio receiver), thermopiles and servo outputs (elevator/aileron, engine). Instead of using a seperate engine controller, the SX board puts a PWM signal out. This signal is powered by a simple MOSFET switcher for direct engine (dc motor) control.

4.2 SX28 chip structure

In opposition to many other microcontroller families the SX chip is optimized for speed and minimized for silicon chip size. In the result the internal chip stucture is very simple. Only one internal peripheral unit exits, the RTCC (real timer clock counter). There is also only one interrupt vector existing.
On the first view this doesn’t look like a successful controller design. But the core runs very fast, at a clock speed of 50MHz the cycle time is 20 ns; remember some years ago the standard TTL logic has an transition time of typical 10 ns (only twice of the controller command speed!).
The solution for the poor internal hardware is the so called virtual pheripheral (VP). That means, that a lot of the real time work of the program has to be done in the interrupt routine.

For the "standard" user it looks a little bit strange when the interrupt routine does much of the progam functions. There are two excellent books [3], [4] about programming secrets of the SX28 chip.

4.3 Real time interrupt

All of the real time functions have to be done in the interrupt. Let's have a look at the necessary functions. There are three output servo channels, three servo pulse input lines, PWM output channel, timer tick, 32kHz auxilliary clock for the optional air pressure sensor and last but not least a failsafe function in case of radio receiver malefunction. That seems a lot of work for one interrupt, doesn't it?
Let us beginn with the servo output signal. As you know a standard servo is controlled by modulated time pulses. The neutral position is reached by generating about 1.5 ms pulse (the exact value depends on the used servo). Maximum and minimum position are limited by times between 2.1 ... 0.9 ms.

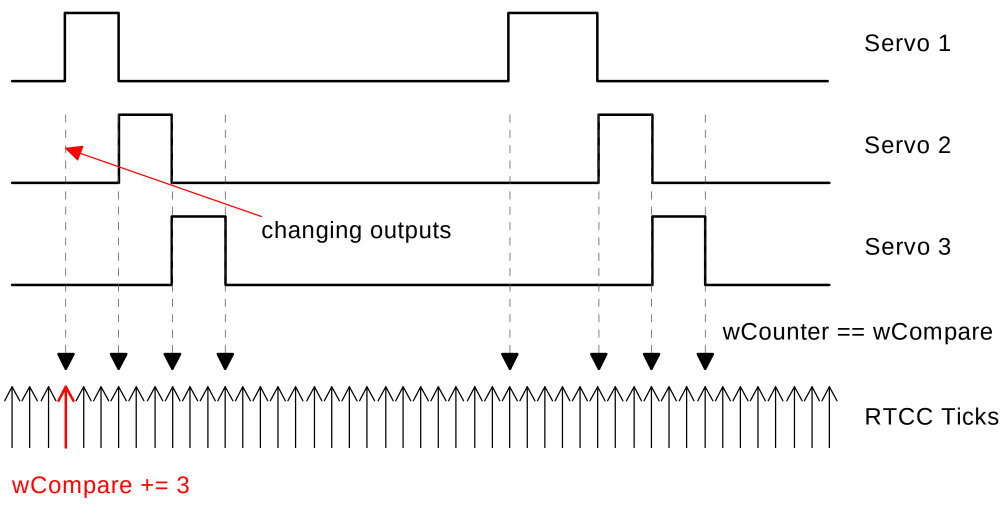


Figure 8 Emulation of a capture compare unit

Figure 8 shows the timing sequence of three independend servo channel outputs. For realisation a capture compare unit will be emulated.
The RTCC runs with a cycle time of 5.12µs. Every interrupt checks the 16 bit counter value (wCounter). If the compare event is true (wCounter == wCompare) the servo state machine switches. A new compare value is calculated and the interrupt ends. Look at the example for better illustration:

```
/* code snippet for servo controlling */
    if(wCounter == wCompare){                  /* generate the servo impulses */
        switch(stIsr.Out){
            case enServo1:
                bServo1Out = 1;
                wCompare += nServoBasic;        /* min pulse time */
                wCompare += bServoOut[enServo1];/* real position */
                stIsr.Out = enServo2;           /* next servo */
                break;
            case enServo2:
                bServo1Out = 0;
                bServo2Out = 1;
                wCompare += nServoBasic;
                wCompare += bServoOut[enServo2];
                stIsr.Out = enServo3;
                break;
            case enServo3:
                /* dito */
                break;
            case enServoPause:
                bServo3Out = 0;
                wCompare += nServoPause;        /*  */
                stIsr.Out = enServo1;
                break;
            }
        }
```

On the same way the timer tick could be generated. In the example a timer tick of 1 ms is needed. The time is short enough, that a 8 bit value (195 x 5.12 µs) will be decreased. If the value goes zero the timer flag will be set and the value is reloaded.

But we don't have an input capture to measure servo input pulses from the radio. It will be possible to emulate such an equipment using the multi input wakeup pins. In this case an interrupt is generated if the specified pin level changes. The interrupt program checks the source and branch.

An easier way is the summation of the logic "1" time of the designated servo channel. Because of the dependencies of the servo channels, it is only one channel active at the same time, an input channel state machine can be designed.

During the first experiments I found that the used receiver produces wrong servo pulses if the radio was powered off or there are large distances between radio and receiver (overrange). In both cases fatal crashes of the aircraft are highly likelihood.

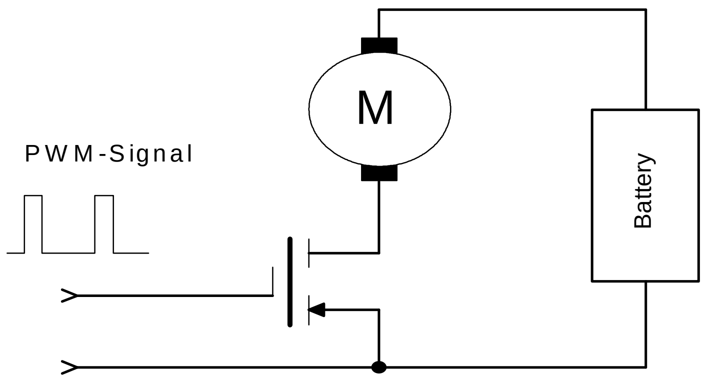


Figure 9 Powering the aircraft engine

A failsafe function has to be implemented. Together with some timeout conditions a "success" flag will be set if the 3 channels are received correctly. Only if this flag is set, the measured servo pulse times are copied from the raw data array to the input data array. If the timeout counter overflows because no succes flag will be set, default values are copied into the input data array. The last function of the real time interrupt is generating a PWM signal for the engine. Normal RC models use a separate motor controller, which "translates" the servo pulse into variable motor revolutions. We replace such an equipment by a piece of

software. The outgoing PWM signal will be used for switching a power MOSFET shown in figure 9.
The MOSFET is directly connected to the dc motor terminals. It is highly recommended to use a component with a 3.3 V compatible digital input (gate), i.e. IRL3402 (International Rectifier).

4.4 Main-Loop

All the cyclic running functions are located in the main loop. The main loop checks the timer tick and starts the cyclic functions depending on their time slot.

```
void main(void){
    DDR(PORTB,OOOOOIIO);                /* PORTB as output, excluding analog comperator */
    /* initialising */
    while(1){                           /* main loop */
        while(!stIsr.Timer);            /* 1ms */
        stIsr.Timer = 0;
        vFccMain();                     /* FCC state machine */
        }
    }
```

The basic software version can be compiled with the CCS compiler by Börn Knudsen (not really recommended, look at chapter 6). Because of the code size limitation (only 1500 words) and the limited math functions the main loop includes only the servo mixing function and the flight stabilisation. Air pressure sensor and radio link aren't used. Especially the simple math function of the free version made it difficult to calculate the air pressure values. An extended software version with the ByteCraft compiler is under development.

5 Hardware design and first flying experiences

At first the model kit has to be mounted. There are an excellent illustrated manual with all building steps. After finishing the aircraft the flight control computer has to be built in.

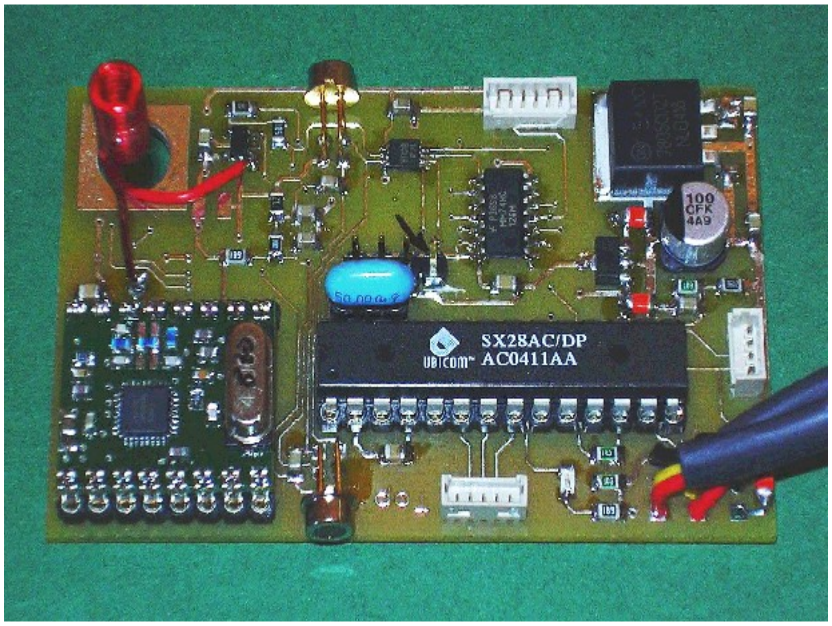


Figure 10 Prototype of the FCC1

A small pcb holds all the hardware components. Figure 10 shows the first prototype. The SX28 is located in the center, above and below the two thermopiles (in the aircraft they view left and right) and on the left the optional radio link.
The FCC1 board is built into the cockpit. All extra componets have to be located very carefully, so that the correct centre of gravity of the model aircraft will not be moved.

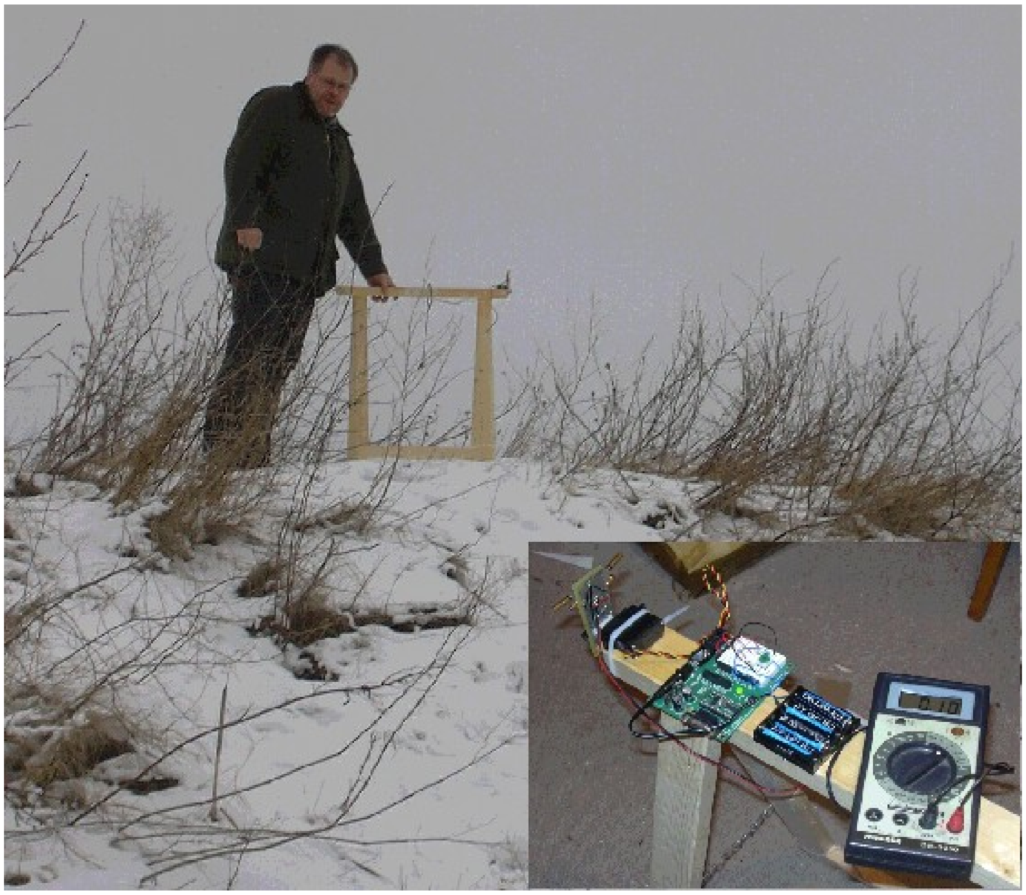
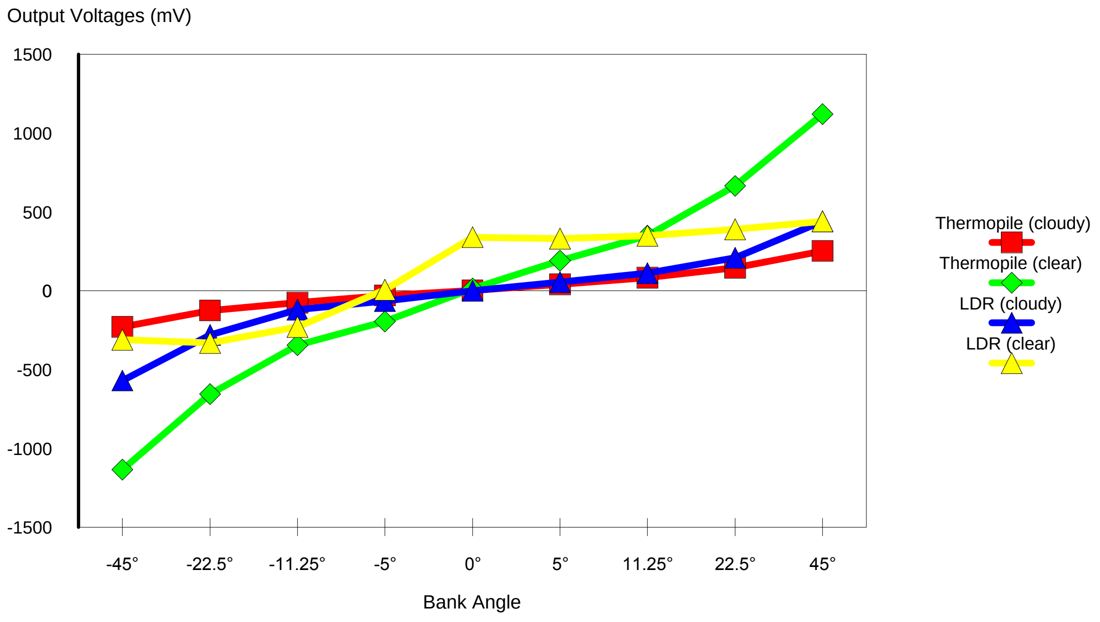


Figure 11 Field application test equipment for horizon detection

Before starting with our flying test, we have to check if the theoretical horizon detecting circuit works in reality. The breadboard thermopile circuit is mounted to a special measurement equipment (figure 11). The complete board is moved by a servo (control signals for the servo are generated by a BASIC stamp).
For simulation and avoiding ground effects the equipment stands on a plate area (top of a flat hill). Then the servo has to be positioned on a set of predefined bank angles.
The results of these measurements are shown in figure 12. The first test was made under bad weather conditions. The temperature was about 3° Celsius and it was very cloudy. In the result of the measured values there is a maximum difference of 70 mV (-30 ... 40 mV) in the bank angle area of +/- 5°. The difference increases under clear weather conditions, difference voltage is now 385 mV (-195 mV ... 190 mV).

With the help of two normal LDRs (LDR - light depending resistor) I checked if larger voltage values can be generated. Unfortunately, the effect of blinding the LDR through direct sunlight was so massive, that this solution fails.
The next problem is detecting bank angles out of the range of -5...5 degrees. It is really difficult to define the switching levels. Small levels produce a lot of sensless servo commands, too large levels would do nothing under poor weather conditions.
As a quick and dirty solution the thermopiles have to be calibrated before starting the aircraft. In this case the model is turned with a low speed about the roll axis. The software checks maximum and minimum voltage levels, so that the right decicions can be made.
To avoid calibration more thermoplies are necassary. For example four components are looking left, right, up and down. Now the different temperatures of ground and sky can be measured, so calibration isn't neccessary. But that's a future project [5].



	-45°	-22.5°	-11.25°	-5°	0°	5°	11.25°	22.5°	45°
🟥 The	-230	-125	-75	-30	3	40	83	145	252
◆ The	-1135	-655	-345	-195	15	190	350	665	1120
▲ LDR	-570	-280	-120	-65	0	55	112	210	440
△ LDR	-310	-330	-230	5	340	330	350	390	440

Figure 12 Output voltages of the horizon detector

Note the yellow points, the simple horizon detector based on two LDRs in opposite directions. During measuring the LDRs were blinded by direct sunlight. The thermopile would not be effected under the same conditions (red, green lines).

6 Conclusion and Future Work

Now, all ingredients for a flight control computer are ready to use. Before the model aircraft should be built, the hardware of the flying control computer has to be tested. For controlling the aircraft a simple radio with at least three proportional channels are enough. A comfortable computer radio with programmable mixing functions isn't necessary because the mixing function is part of the software of the FCC.
The radio receiver is connected to JP5 (figure 2). Make sure that the servo pulses from the radio are in chronological order as shown in figure 8. The first servo signal from the receiver must be the aileron control signal und the second the elevator. The mixer combines both impulses to the special needs of the model. The third channel controls the engine.
After power connecting the red LED flashes. There are two flashing frequencies. Low speed flashing signalises that the pilot controls the servos, while high speed flashing the lateral axis is controlled by the thermopile signal. Elevon and engins are not influenced in this case.
If all control commands are received correctly, the first flight could be started. Good luck!

The project gives you an idea to make your own intelligent flying equipment. With the help of the software example you get the chance to build your own "SAM". Originally I tried to use a free C compiler, i.e. the CC1B from Knudsen Data, but unfortunately I can't recommend this version. The version 0.6B is awful.
The basic software was compiled by the Bytecraft C compiler. Special thanks to Walter Banks from ByteCraft for his support during developing the project. The software is as simple as possible. Theoretically, you can compile the source with the free CC1B with only a few modifications (set the compiler switch #define CC1B).
I can imagine, that it should be also possible to use the Parallax BASIC compiler. The interrupt service routine has to be programmed in Assembler, but the other software could be written in BASIC.
The implemented stabilisation algorithms are as easy as possible. The calibration was replaced with a comparison with fixed values (because of the contest’s deadline). The problem of safe horizon detection is so complex, that it should be a good idea to spend a separate controller to the thermopile sensors. With better mathematical algorithms I'd expect improving flying behaviour.
It should also be sensful to use more than two sensors. In case of using at least five thermopiles (up, down, left, right, forward) bank angles can be measured.
But let's have a look at the results. The futuristic swept forward aircraft is controlled by a simple hardware including all of the (servo) mixing parameteres, a failsafe function in case of overranging and the integrated PWM controller for the engine.
The hardware is prepared for a set of options, like radio link or air pressure sensor. A serial I/O port allows to connect other components. In case of using all these options it is recommended to choose a larger microcontroller, like SX52 etc.
The original pcb data contains the file 4eq2_pcb.gwk. This file can be used for ordering from the pcb manufacturer Beta-Layout. Please download the pcb viewer Gcprevue.

Note there is one modification necessary. Pin 3 of JP7 has to be connected to 5V for programming with the SX-Key. Open the existing connection to 3.3V and connect pin 3 to 5V potential with a short wire.

7 References

[1] Taylor,B; Bil,C.; Watkins,S.: "Horizon Sensing Attitude Stabilisation: A VMC Autopilote", 18th International UAV systems Confrence, Bristol, UK, 2003

[2] Lennon, Andy: "R/C Model Aircraft Design - Practical Techniques for Building better Models", Published by Air Age Inc. 1996

[3] Williams, Al: "Beginning Assembly Language for the SX Mikrocontroller", Parallax Inc.

[4] Daubach, Günther: "Programming the SX Microcontroller - A Complet Guide", Parallax Inc.

[5] Altenburg, Jens: "TOPAS - Thermo optical attitude detector for an UAV" PSoC - Design Contest 2004, www.circuitcellar.com/psoc2004

8 Appendix A - PCB Layout

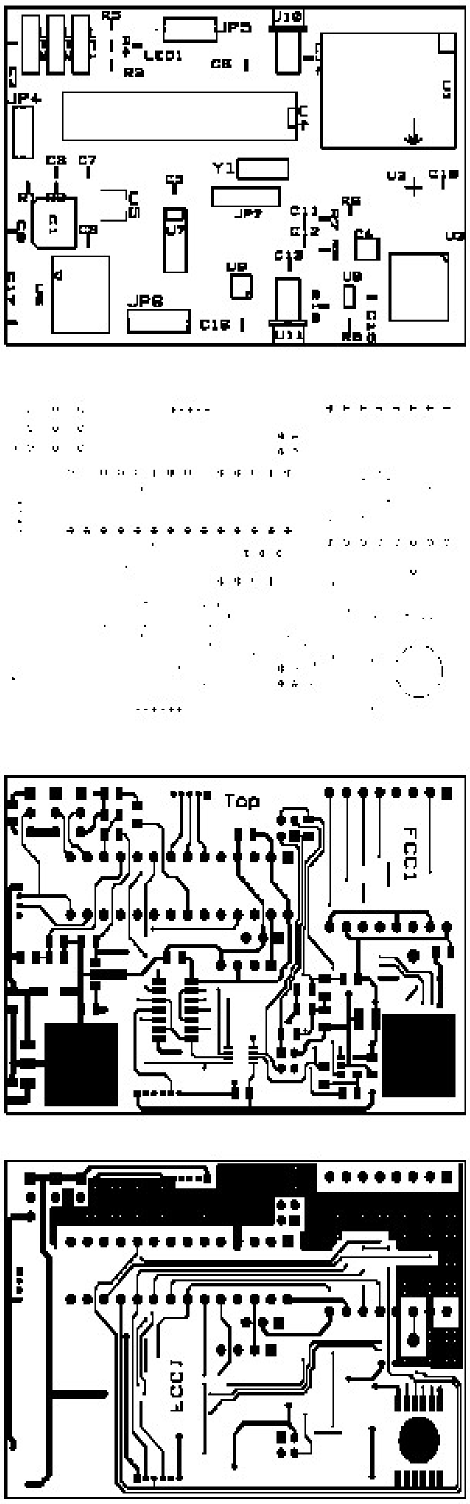


Figure 13 PCB, data available in GWK format for Beta-Layout (www.pcbpool.com)

9 Appendix B - PCB Bill of Material

Item	Quantity	Reference	Value	Pack./Manu.
1	1	PCB (file: 4eq2_pcb.gwk)		www.pcbpool.com
2	1	C1	100u	Panasonic
3	3	C2,C7,C17	10u	SMD 0805
4	12	C3,C5,C6,C8,C9,C10,C11, C12,C13,C14,C15,C16	100n	SMD 0805
5	1	C4	47u	Tantal Size B
6	3	JP1,JP2,JP3	Header 3	
7	1	JP4	53047-0410	Molex
8	1	JP5	53047-0510	Molex
9	1	JP6	53047-0610	Molex
10	1	JP7	Progamm Connector	
11	1	LED1		SMD 0805
12	1	R1	36k	SMD 0805
13	3	R2,R3,R4	10k	SMD 0805
14	1	R5	680R	SMD 0805
15	3	R7,R8,R9	1k	SMD 0805
16	1	R10	1M	SMD 0805
17	1	U1	TRX905	Option
18	1	U2	Antenna	
19	1	U3	M5534	Intersema
20	1	U4	SX28AC	Parallax
21	1	U5	XC62FP500	SOT89
22	1	U6	MC7805CD2T	TO263
23	1	U7	74HC126	14SOP150
24	1	U8	AD8628	SOT23
25	1	U9	MAX144	Dallas
26	2	U10,U11	10TP581T	Semitec
27	1	Y1	50MHz	Parallax