A MATCHBOX CHESS COMPUTER

Harm Geert Muller

Amsterdam, The Netherlands

In 1985 I was invited by the Computer-Chess Association of the Netherlands (CSVN) to compete in their fifth national championship with my old chess programme "USURPATOR II". I had enlisted in that tournament some years before, but had long since lost the 6502-based computer (a Rockwell AIM '65) on which I had developed the programme. It would have been hard to find a similar system that had its operating speed boosted by a factor of two, like the one I had owned. Furthermore I had very unfavourable experiences with transferring computer programmes stored on audio tape to other computers with incompatible tape formats, and I was sure the only tape I had left containing the source programme was incompatible with anything else, because I had designed the tape format myself to circumvent the slow and error-prone cassette interface of the AIM '65.

I therefore decided to build a new computer for the occasion, one that would run as fast as I could make it. I did not want to spend too much time on this project, and so I wanted it to contain as few parts as possible. Since USURPATOR II consisted of only 4 Kbyte of code and used about 1K for data storage, a single 8K memory chip would be sufficient to hold both the programme and the data. I did not want to store the programme in EPROM, because in those days EPROMS where fairly slow memory devices. Therefore I decided that the new computer should consist of a fast 65C02 CPU chip, an 8Kbyte static memory (RAM) chip and whatever was needed to support those.

After some thinking it became clear that the volume of the components would hardly total more than the volume of an ordinary-size matchbox. It next became a challenge to squeeze the entire engine indeed into such a matchbox. Since in those days the technology of 'surface mounted devices' was not very developed yet, the use of chips in common 'dual in line' packages was mandatory. Especially the CPU chip wasted a lot of space because of its bulky plastic encapsulation. I had even to file a few millimeters off the ends to make it fit at all.

Cannibalizing a Calculator

Input and output is of course a problem in a computer the size of a matchbox. Of course I could have equipped the matchbox with a single LED that would flash the selected move in Morse code. Since I don't know Morse code at all, this idea seemed too cumbersome. The next best thing I could think of was a very tiny seven-segment LED display like those used in old pocket calculators. Liquid crystal displays used in watches would be smaller still, but after demolishing a watch I learned that every segment in such a display has to be driven separately, thus requiring a prohibitively large number of outputs on the computer. So I sacrificed my old pocket calculator to obtain a three-digit LED display.

This display required only seven output lines to drive the segments, while output line number eight drove a transistor that selected the digit of which these segments were on. The third digit was not used, and the move could be displayed by alternating the algebraic notation for the original and the target square in the two available digits. As is common in displays of this type the impression is created that the two digits needed to represent a square are lit simultaneously by switching 100 times a second between displaying the letter in the left position and the digit in the right position. The display could thus be controlled by an eight-bit output port, a function that could be fulfilled by a single chip. Displaying something requires the continuous attention of the computer, and has to be stopped while the chess program is thinking.

The input problem was much harder to solve than the output one. Even the smallest switch I could buy in my local electronics store would take up one quarter of the matchbox. Touch buttons could be made smaller than that, so I opted for those. The question was how many would be needed to operate the machine. Since it was clear that I did not have space to assign a key to each of the letters A to H and digits 1 to 8, some kind of sequence would be needed to enter the moves. If a sequence was needed anyway, multiple keys would only serve to confuse matters, so I decided to have just a single touch contact on the top surface of the matchbox. It consisted of two staples put into the cardboard, that could be connected electrically by putting one's finger on top of them.

What Happens to Stapled Input

To include a separate chip for just a single input line seemed wasteful, and therefore the touch contact was connected directly to the interrupt line of the CPU chip. This causes an interrupt each time the contact is touched, and software can determine how much time elapses between interrupts, and so decode the meaning of the sequence. It later turned out that the touch contact did not operate very well in a dry environment, and an extra transistor was added to amplify the signal. The drawback is that this made it perform poorly in a wet environment, so that one day after biking through the rain with the matchbox in my pocket I had to use a hairdryer before I could play. But I knew you couldn't win them all.

Eventually the design contained only five chips (CPU, memory, output, and two chips with gates used for miscellaneous functions), a display, two transistors, two tiny batteries, and a 16 pin IC socket. The socket can be used to plug in a cable connecting the matchbox to a personal computer, that then downloads the memory chip with software. Virtually no space is left inside.

Entry Just in Time

The entering of moves through the single contact seems very awkward at first, but after a dozen moves or so becomes quite natural. One simply taps on the touch contact a number of times in rapid succession, once for A or 1, twice for B or 2, etcetera. Between the letter and the digit one simply waits somewhat longer. To aid the operator, the character being entered is displayed, so after touching the contact once an A appears in the display, and if we touch again before the A disappears (after 300 ms) it changes into a B, and so on up to H after which it wraps around back to A. If the desired character is in the display we wait until it disappears, and start tapping again for the next character. Of course the computer now knows that we are entering a digit, so it counts 1, 2, 3, ..., up to 8. If we do not touch the contact for longer than a second without having entered four characters in this way, the move is void and we have to start from scratch. The computer then goes back to displaying whatever it was displaying before we started touching it. This seems a harsh way to treat the operator, but is important to prevent an accidental touch from erasing the move selected by the computer from the display.

If four characters are entered this way (requiring at most 31 touches, e.g. for h7-h8) the move is flashed in the display for three seconds, for final scrutiny on the part of the operator. If the move turns out to be in error after all, starting the entry of a new move within this period will overwrite the old one. After the three seconds have elapsed, the display is erased, the move is tested for legality, and depending on this either a prompt sign or an error symbol is displayed.

USURPATOR never starts thinking of its own accord. To have him play a move one has to press the contact for one steady second. The A in the display then changes into Ξ, and if we release the contact at that moment, USURPATOR starts thinking for the player whose turn it is to move. This procedure makes it possible to enter a sequence of moves to set up a game, play a game with either Black or White against the computer, or have the computer play itself, without the need for any other commands. One can even change sides halfway a game.

Don't Watch Us, We'll Beep You

After a few matches in the championship it became clear that there were two major shortcomings to this user interface. The first was that it required constant attention to know when USURPATOR made his move. One had to watch the tiny display like a hawk to avoid losing time. This was solved by including a beeper disk from a watch into the design. This very flat disk could be fitted between the drawer and the wrapping of the matchbox, at the bottom. It was connected to an unused address line of the CPU, so that the programme could change the voltage on the disks by jumping back and forth between two memory addresses for which this address bit differed. By doing this in rapid succession an audible beep is created.

The other problem was that it was very cumbersome to set up a previous position if an entry error had occured. The rules of the tournament are such that if one of the operators made an entry error that went unnoticed for some time, the game should be restarted at the position just before the error was made. The only way to do this was to interrupt the power to reset the matchbox, and then enter all the moves played so far. To facilitate this, the matchbox now stores all the moves played so far. If the contact is touched extremely long, the display starts counting up from zero while the touch lasts. If the contact is released while the count reaches n, the game is restored to the situation after move number n. This also makes it easy to start a new game, by simply releasing the contact when the zero appears.

The least one can say about USURPATOR II, playing from its matchbox, is that the program is lightning fit. Not quite so fighting fit because it is not a very strong programme. The reason for this is that it started as an experiment, and this experiment was never really completed. USURPATOR relies very heavily on a Shannon-B strategy. To decide which moves are considered while searching the game tree, it distinguishes several levels of plausibility for moves. This plausibility does not in any way depend on an estimated value of the move, (which is determined by the tree search), but rather on how well the move logically follows from the previous moves.

For instance, if a piece moves to a certain square, it is logical to try and capture it on this new square. This does not mean that it is always good to capture it, but we can never know that before we try. Other logical continuations would be to try and capture something with a move that passes over the square just emptied (after all the piece might have been pinned down), or moves that save a piece that was attacked at the last ply.

The Penalties for Off-side

Moves of this type are always considered by USURPATOR, one could think of them as being part of an elaborate quiescence search. If they occur early in the tree they are not counted towards tree depth either. Only on the first two plies is a Shannon-A strategy used, because the programme is not supposed always to remember what went on before the current position was reached. The first two plies thus act as initiators of a plan, and after that only moves consistent with the plan are considered, on the assumption that all other moves are irrelevant to the point of nonsense.

In some situations USURPATOR's strategy works extremely well, but in others it fails miserably. The reason is that the strategy outlined above is not really implemented very consistently. For instance if Pawns are attacked this is considered such a minor threat that moves that could save them are not considered logical. The most severe shortcoming is that USURPATOR has only a very approximate notion of how to defend its pieces. It considers defending a valid way to fence off an attack from the previous ply, but if a move leaves certain pieces undefended this goes completely unnoticed. Furthermore it is easy to throw USURPATOR's analysis off track by an intervening exchange, after which it has forgotten all about its original plan.

Some of these problems could be fixed easily, but for solving others a drastic overhaul of the entire programme is desirable. I have therefore decided to write a new programme from scratch, and I might as well do this for another machine, since the matchbox computer is an environment that is not well adapted for programme development. I now use the matchbox only to entertain the spectators on special occasions, such as the tenth re-enactment of the Dutch National Computer-Chess Championship. Although the programme has not been changed in 7 years, sometimes it still surprises me by actually winning a game. Among its major achievements are victories over an 80386 and an 80286 machine in 1990, and a draw against a Gould computer in 1985.

Collection of Photos by Jos Uiterwijk

FIREPOWER FOR A FULL BOARD. For author Muller a matchbox packs enough kindling to set a match ablaze.