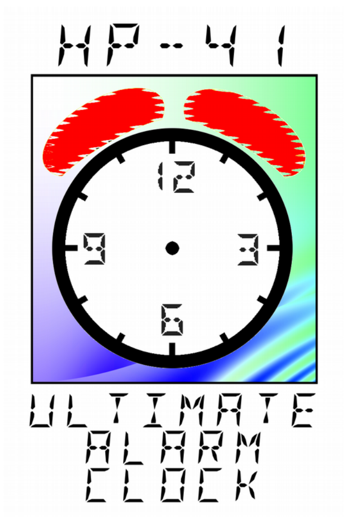
The ISENE Rom (IS-2B)

This 4K module image contains useful time management, file management and utilities software for the amazing Hewlett Packard 41 calculator. All programs described here are in FOCAL. One MCODE utility is included and used by some of the FOCAL programs, “CURXMFL”. It adds the current XM ascii file to Alpha. Requirement: An HP-41CX (or equivalent, such as an HP-41CL)

Table of Contents

UAC (the Ultimate Alarm Clock)



The program that turns your HP-41CX (or 41C/CV with a time module) into the ultimate alarm clock. It gives your beloved calculator all the features of a very advanced alarm clock. It even lets you hear the time so you don't have to turn on the light to see what time it is :)

I have a passion for old Hewlett Packard calculators. I love to program these little beauties. Now and then I even make a very useful program such as this one. I have never seen an alarm clock that can do everything I wanted it to. So, I decided to program my HP-41CX to serve my needs. It has the features I want:

- ✔ Set the alarm to a specific time
- ✔ Set the alarm to a certain time from now
- ✔ Set the alarm to a preset time from now
- ✔ Move the alarm a ½ hour into the future
- ✔ Move the alarm anytime into the future
- ✔ Hold the alarm clock until you restart it
- ✔ When the alarm goes off, stop it by pressing any key
- ✔ When the alarm goes off, snooze by ½ hour
- ✔ Double-alarm (autosnooze); Alarm goes off, then again 10 minutes later
- ✔ Clear the next alarm
- ✔ Clear all alarms
- ✔ Hear the time!
- ✔ Hear when the alarm is supposed to go off!
- ✔ Hear how much time left until the alarm goes off!
- ✔ Set the alarm for each day throughout the week (daily alarms)
- ✔ Stop or restart the daily alarms

Here is the key mapping (what shows in the programs menu is in parenthesis):

Label	Description
UAC	Starts the program. Shows some of the mapping for the top keys so that you don't have to remember this list. Pressing R/S will give you the running clock. Pressing R/S again will give you the alarm catalog, then the version number of the UAC, then it returns to the mapping/menu. It may be useful to assign this function to TAN.
ALM	Set an alarm. Just enter the time and execute ALM. If the time greater than the present time, it will set the alarm for that time later today. If the time is less than the present time, it will set the alarm time to that time tomorrow. It may be useful to assign this function to SIN.
AL+	Enter the time from now you want the alarm to go off and execute ALM+. It may be useful to assign this function to COS.
LBL A (.5)	Postpone the alarm by ½ hour (both if the alarm is sounding or if it has not yet sounded). If the "wake-uo"-type alarms have been cleared (by pressing "C" or "c"), it will set an alarm ½ hour from now.

LBL a (+)	As above, but postpone the alarm by the amount you enter.
LBL B (8)	Set the alarm 8:10h into the future (to change this preset time, change line 28 in the program).
LBL b (9)	Set the alarm 9:00h into the future (alter by changing line 35)
LBL C (c)	Delete the next "wake-up"-type alarm.
LBL c	Delete all alarms. Execute WKR (see below) to reactivate the daily alarms.
LBL D (A)	Hear how long it is until the alarm goes off. Same principle as with SCLK - one low beep per 6 hours, then one high beep per hour, then one low beep per quarter. If the alarm has already sounded, it will first give a BEEP, then tell you how long since it went off (same principle).
LBL d	Hear when the alarm is supposed to go off. Same principle as with LBL D.
SCLK/ LBL E (T)	Hear the time. First it sounds 1-4 low beeps signaling which sector of the day the time is within: 0:00-05:59, 06:00-11:59, 12:00-17:59 or 18:00-23:59. Then 1-6 beeps signaling the number of hours within the sector, then 1-4 beeps signaling which quarter it is within the hour.
LBL e	"Hold" the alarm until you press R/S to let it continue again. This makes it possible to wake up in the middel of the night with a brilliant idea that you want to write down. Press "e" to hold the alarm until you are ready to return to bed. Then press R/S (or "e" another time and you still get your usual hours of beauty sleep.

When the alarm goes off, press any key to stop it. If you then press R/S, it will snooze for 10 minutes.

Set flag 2 (FS 02) to make the alarm "autosnooze" by automatically sound again 10 minutes after it first goes off. This is also a useful feature if you are hard to wake up.

The UAC gives you sound feedback for all the functions. This is done so that you know if you pressed the right button in the dark.

You may also want the WKA (WeeK Alarm) program as an add-on module to UAC. The program is made as an add-on because it is not part of the UAC core functionality and requires X-Functions and X-Memory. It is therefore not of value to those with a 41C(V) and a time module.

WKA makes it possible to set specific alarm times for every day throughout the week. First it will tell you that the week starts with Sunday (SUN) as day 0. Then it will ask for the alarm time for each day starting with Sunday. By entering 0, no alarm is set for that day. It may be useful to assign WKA to TAN-1.

To stop the daily set alarms, execute WKS (Week-alarms Stop). It may be useful to assign this function to SIN-1. To restart the daily alarms, execute WKR (Week-alarms Restart). It may be useful to assign this function to COS-1.

TIMER (sound alarm every X minutes)

A cute and useful little timer program.

It will sound two short, high tones every X minutes.

Simply enter the interval in minutes into X and do XEQ'TIMER and the calc will turn on every interval, sound the two high tones and turn off again.

This program is helpful if you want to keep tab on for instance every 10 minute interval.

D-W & W-D (Date to Week# & Week# to Date)

This is a simple date-to-week number-to-date converter.

The program converts a date to the ISO week number and a week number to the starting date of that week (Monday as specified by ISO). It takes the date in any of the HP-41 formats. The week number to be converted to the date (Monday) can be WW for the current year or WW,YYYY for any year.

The program is actually two programs in one; the D-W and the W-D.

Examples:

Enter (in DMY mode):

2,012006
XEQ'D-W
and you get:
1

Enter (in MDY mode):

11,2007
XEQ'W-D
and you get:
3,122007

Enter (in MDY mode):

11
XEQ'W-D
and you get:
3,132006

REM (REMembering – event management)

Now your HP-41 can truly come to everyday use. Move beyond the 10 alarms limit of the calculator. Let your calculator take care of remembering and keeping track of appointments. Features include:

- Very easily enter new appointments
- List all appointments. If you have a printer, let it print them out
- Quickly edit appointments
- Show today's events

When first run, the program will create an XM file named “REMF” to store all appointments.

Here is the key mapping (what shows in the programs menu is in parenthesis):

Label (Menu)	**Description**
REM	Starts the REMembering program. Shows some of the mapping for the top keys so that you don't have to remember this list. Pressing R/S will give you the version number of REM, another R/S will get you back to the mapping/menu..
R+	Enter the date of the event. Enter the time of the event. In the alpha register, enter the message (up to 12 characters). Then XEQ "R+". This is a global label so that you can assign it to a key for easy entry of new appointments.
LBL A (+)	Same as R+ above, but used inside the program.
LBL a (-)	Delete the current record.
LBL B (L)	List all appointments. The appointments will be shown the way you enter new events, i.e. the alpha content will be active for each appointment shown, with the time in the X-register and the date in the Y-register. If you have a printer attatched, it will print out all the appointments in the storage format (the format used in the extended memory file "R" is "MM,DD:HH,MM:EVENTMESSAGE").
LBL b (E)	Edit the current record (calls ED).
LBL C (?)	Search the event file for string in alpha register.
LBL D (T)	Show the events of today (simply calls ALMCAT).
LBL d (S)	Forces a sort of the event file ("R") so that the events are in proper date/time order. This function is available because new events are added in the beginning of the file and this may not be where it should appear if sorted correctly.
LBL E (*)	Go back to the menu.
LBL e	Initiates the program (sets up the alarm calling ^^R - see below for technical info).

Technical notes:
The program will store the events in the format "MM,DD:HH,MM:EVENTMESSAGE" regardless of the date format you use on your calculator (MM,DDYYYY or DD,MMYYYY). This is done for sorting convenience. When the program is initiated (via LBL e), it sets up an alarm that executes "R" every night. This is where the magic happens. "R" will go through the extended memory file "R" and cull all events for the day and make them into alarms, deleting them in the XM file.

HOURS (hour registration)

For consultants who need to track and register billable hours or for others in need of registering hours worked on projects or tasks.

When first run, the program will create an XM file named “HOURSF” to store all entries.

To use this program, you must create the XM ASCII file "HPROJS" and enter an abbreviation for each project (make it 1 to 3 letters). One project per record in the file. The program will create a custom menu out of these project abbreviations, like "P1 P2 P3 P4", where each abbreviation corresponds to a local label (here A to D). In this example hours to the "P1" project is registered by pressing the A key (in User mode), while hours to the "P3" project is registered pressing C.

To register hours, run the program. The current date will be in the X register. If you want to register hours for another date, then enter that date (only month and day in the format that you have for your calculator - MDY or DMY; Example = Enter in the X register "25.04" for 25th of April if you have DMY as the format or "04.25" if MDY is your format). Enter the hours worked that day. Enter the description in Alpha and press the local label (A up to E) for the project you want to register hours for. The program shows you the record (truncated to 24 characters) that is added to the HOURSF file (the XM ASCII file where all hours entries are registered). If you have your HP-41 hooked to a PC via HP-IL, you may then download and extract all hours from HOURSF by using this solution: https://github.com/isene/HP-41CL_HOURS.rb

In addition to the dynamically created local labels A - E, pressing the local label "a" (or using the global label "PED") will let you edit the HPROJS file. Label "c" (or "HCLR") will clear (empty) the HOURSF file and "e" (or "HED") will let you edit the HOURSF file (in case you have misregistered something).

You may also use the FILE program to edit or print the HPROJS and HOURSF files.

NOTES (quick notes)

No real PDA can do without a quick notes taker.

The same goes for the best PDA ever created: The HP-41.

The simple feature list:

- Simple interface for adding new notes
- Listing notes (or printing them if a printer is attatched
- Editing, sorting and searching

Here is the key mapping (what shows in the programs menu is in parenthesis):

Label (Menu)	**Description**
NOTES	Starts the NOTES program. Shows some of the mapping for the top keys so that you don't have to remember this list. Pressing R/S will give you the version number of NOTES, another R/S will get you back to the mapping/menu..
N+	Enter the note in the alpha register. Then XEQ "N+". This is a global label so that you can assign it to a key for easy entry of new appointments.
LBL A (+)	Same as N+ above, but used inside the program.
LBL a (-)	Delete the current record.
LBL B (L)	List all notes. If you have a printer attached, it will print them.
LBL b (E)	Edit the current note (calls ED)
LBL C (?)	Search the notes file for string in alpha register.
LBL D (S)	Forces a sort of the notes file ("NOTES").
LBL d (/)	Forces shrinkage of the "NOTES"-file..
LBL E (*)	Go back to the menu.

CRYPT (HP-41 cryptography)

Introducing encryption to the HP-41

This cryptography program encrypts or decrypts an ascii Extended Memory (XM). Ypou can also temporarily view an encrypted file without substituting the file with the decrypted version.

The CRYPT program implements the Vigenere cipher with a key of your choice of up to 300 characters. By default it uses the range of characters from ascii code 32 (space) to ascii code 90 ("Z"), but you can choose any range above ascii code 32 - for example a range value of 90 to include lower case characters (ascii code 122 is "z").

If the key is at least as long as the file to be encrypted, you will actually get perfect security for the encrypted file. You will have what is known as the One-time pad

Here are the three functions implemented:

Function	**Description**
ENCR	Encrypts an ascii file. The file name must be in Alpha when you execute ENCR. The program first prompts for the character range (default 58 - press R/S to accept the default). It then prompts for the key (Alpha is set to ON). Enter the key to use for the encryption 24 characters at the time. As long as you fill the Alpha register with characters for the key, it will keep asking for more key - until you enter less characters than 24 - then it will commence to encrypt the file. The program resizes the memory to accomodate for a large key if necessary. After encrypting the file with the key, you get the message "DONE" along with a beep. All traces of the key have then been removed. Pressing R/S again will launch the EDitor (ED) with the encrypted file.
DECR	Decrypts the file (name in Alpha. Prompts for the range and the key (use the same range and the same key as when you encrypted the file. Again you will get a "DONE" and a beep when the process is completed. Pressing R/S again will launch the EDitor (ED) to let you view and edit the decrypted file.
VIEWCR	Lets you view an encrypted file. Instead of actually decrypting the whole file, you get to view each successive record. The program sounds a Tone 9 upon showing each record. The file remains encrypted in Extended Memory. After the last record is viewed, the program displays "DONE" and gives a beep. All traces of the key is removed.

EVAL

Simplifies the evaluation of cases against a requirement specification

A requirement specification consists of a set of requirements or "items". Each item is given a certain importance or "weight".

A requirement specification could be used to decide what piece of production equipment to buy, for choosing a new house or for choosing a new employee. If you are to choose a new employee, the specification would consist of items such as "relevant knowledge", "relevant job experience", "proven production record", "communication skills" or "empathy". You would give each item a certain weight where "relevant knowledge" could be given a weight of "4" while "empathy" for this specific job could be given "2" in weight. It doesn't matter what minimum or maximum you choose for you "weight" scale, only the relative weight scores matters – like "relevant knowledge" being twice as important as "empathy" (their weight scores might as well have been "30" and "15").

When a requirement specification is populated with a set of weighted items, it's time to pitch a set of cases against the specification. A case would be one of the candidates for a job or one of the houses you are looking at buying. A case is given a score on each item. You figure out the scale you want to use and put a score on each item of the requirement specification for the case you evaluate. The scale goes from "0" to any number you set as the maximum score. A candidate for a job could score a "3" on "relevant knowledge" and a "5" on empathy on a scale from "0" to "5". You then multiply the score with the item's weight to get the "weighted score". So even though the candidate receives a maximum score on empathy, she only gets a weighted score of "10" on that item compared to "12" on "relevant knowledge".

Finally, you sum up all the weighted scores, divide by the sum of the item weights, divide by the maximum score and multiply with 100. Then you have a total percentage score of how well that case fits the requirement specification.

The formula for the total percent score is:

$$T\ \% = 100 * \frac{\sum_{i=1}^{I} v_i w_i}{v_{max} \sum_{i=1}^{I} w_i}$$

"T%" is the total percentage score, "w" is the weight, "v" is the score (value) of an item, v(max) is the maximum score possible.

The EVAL program makes all these evaluations a breeze. You run the program (XEQ "EVAL") and are prompted for a Project Name. The program calls each requirement specification "a project". Enter a name, and if the project does not exist as an Extended Memory file, a file is created and you are asked first for the maximum score you will use when evaluate the cases (the top of the scoring scale). Then you are asked to input the items, one by one using the format "WEIGHT:DESCRIPTION" in the Alpha register (such as "2:EMPATHY"). When you are done entering the items, just press R/S without entering anything on the prompt, and you are taken to the main menu. If a project file already exists, the program will pull that file up and take you directly to the main menu.

From the main menu, you can choose to add more items to the requirement specification, List the items in the file, EDit the file, or decide to enter a case with scores for each item in the requirement specification. Pressing "E" will always take you back to the main menu.

When you press "D" to enter a specific case, Flag 01 is set and the main menu changes. The program uses a "dynamic menu" to change labels A-E depending on whether you are handling the requirement specification or handling specific cases and evaluate them against the specification

The main menu for the cases include the ability to List the scores given on each item (or indeed change the scores already given), to evaluate the case against the requirement specification to yield a total percentage score, to enter another case or to go back to handling the requirement specification (Flag 01 is then cleared and the other menu is again activated). Again, pressing "E" while working with cases will get you back to the "case menu".

You may at any time press "d" to delete (purge) the current file you are working on, whether it is the requirement specification (the project file) or a case that you are working on.

Here is the key mapping (what shows in the programs menu is in parenthesis) when working with a project file (the requirement specification):

Label (Menu)	**Description**
EVAL	Starts the EVAL program, asks for Project Name. If it exists, pulls that file into memory and takes you to the main menu for working with the project file. If a project file with that name does not exist, creates the file, asks for the maximum score you will use when giving scores for cases and then takes you to LBL A to input the items for this requirement specification.
LBL A (I+)	Add an item to the requirement specification The program asks you to enter each item with the weight (a number), followed by a colon (":") and then the item’s name/description. When you are done entering items, simply press R/S without inputting anything and you are back at the main menu.
LBL B (L)	List all the items in the project file.
LBL C (ED)	EDit the project file. Note that the maximum score is saved in record 00 of the project file.
LBL D (C+)	Enter a case to evaluate. The program prompts for a score for each item in the requirement specification. When all the items have been given a score, it calculates and displays the total percentage score for that case (how well the case fits the requirement specification as a total percentage).
LBL E (*)	Go back to the project menu.

While a project file is saved as an Extended Memory ASCII file, a case file is an Extended Memory DATA file. Here is the key mapping (what shows in the programs menu is in parenthesis) when working with a cases:

Label (Menu)	**Description**
LBL A (C+)	Add another case to the project. (Same as LBL D from the project menu).
LBL B (L)	List all the scores for each item for the current case. You may proceed to change each score.
LBL C (W*V)	Calculate the total percentage score for the current case against the current requirement specification.
LBL D (^)	Move back up to working with the project file (takes you to the project menu).
LBL E (*)	Go back to the case menu.

Register setup:

REG 00 scratch

REG 01 PROJECT NAME

REG 02 SCALE

REG 03 # OF VALUES

REG 04 ITEM NAME

REG 05 ITEM VALUE00

REG 06 ITEM VALUE01

REG … etc.

File management system

Here are two menu-driven programs for easy file management on the HP-41. While "FILEMAN" handles creation, deletion, clearing and save/get to/from HEPAX memory, "FILE" handles the contents of ASCII files. These programs requires the LIBXM and LIBHPX libraries. The function XMFILE? in the LIBXM will be skipped unless you have an HP-41CL. It is included for the added convenience of showing the user what file is currently handled at the start of these programs.

FILEMAN

The program shows the short form of labels A-E and then a-e as prompts:

- **"C:A D G:A D SK"**
- **"P CL S:A D FL"**

- LBL A: **C:A** = Create ASCII file (name in Alpha)
- LBL B: **D** = Create Data file (name in Alpha)
- LBL C: **G:A** = Get/retrieve ASCII file from HEPAX memory (name in Alpha)
- LBL D: **D** = Get/retrieve Data file from HEPAX memory (name in Alpha)
- LBL E: **SK** = Select file (name in Alpha) and set record to 0 (i.e. execute a SEEKPTA)
- LBL a: **P** = Purge file (name in Alpha)
- LBL b: **CL** = Clear file (name in Alpha)
- LBL c: **S:A** = Save ASCII file to HEPAX memory (name in Alpha)
- LBL d: **D** = Save Data file to HEPAX memory (name in Alpha)
- LBL e: **FL** = Run the "FILE" program (see below)

FILE

The program shows the short form of labels A-E and then a-e as prompts:

- **"+. .+ +1 +X ED"**
- **"- .+a -1 S ><"**

- LBL A: **+.** = Insert record (in Alpha) before current record
- LBL B: **.+** = Append record (in Alpha) after current record
- LBL C: **+1** = Jump one record forward
- LBL D: **+X** = Jump the specified number of records (in X) forward (or backward if X is negative)
- LBL E: **ED** = Edit current file
- LBL a: **-** = Delete current record
- LBL b: **.+a** = Append Alpha to current record
- LBL c: **-1** = Jump one record backward
- LBL d: **S** = Sort file alphabetically
- LBL e: **><** = Trim file (i.e. run FLSZ- from LIBXM)

Libraries & utilities

Essential libraries for handling of XM and HEPAX files

LIBXM

This library contains functions for handling eXtended Memory files:

- FLSORT: Alphabetically sorts an XM file (file name in alpha)
- FLSZ+: Ensures the XM file (name in Alpha) has room for 4 more records
- FLSZ-: Trims the XM file (name in Alpha)
- SKPTACR: Sets the file pointer of file to 0. Creates file if needed
- XMFILE?: Return the current XM file name in Alpha.
  This function requires the an HP-41CL to actually execute.

LIBHPX

This library contains functions for handling HEPAX files:

- HSAVEAS: Saves XM ASCII file (name in Alpha) to HEPAX memory
- HSAVED: Saves XM data file (name in Alpha) to HEPAX memory
- HGETAS: Retrieves HEPAX ASCII file (name in Alpha) to XM
- HGETD: Retrieves HEPAX data file (name in Alpha) to XM
- HRESZFL: Resizes HEPAX ASCII file (name in Alpha) to specified size (in X)

DMD

This little function switches a number between the formats dd.mm and mm.dd (where “dd” is the day of the month and “mm” is the month number).

RNG

Returns a random number between 0 and 1. This Random Number Generator (RNG) uses TIME as input, rendering the numbers generated “more random” than the usual RNGs that keeps a seed stored for the next value generated.

EASTER

Enter the year. The program returns the date for Easter Sunday in the format YYYY.MMDD (the answer for 2018 XEQ”EASTER” would be 2018.04.01, i.e. April 1st.

CJC (Calendar date – Julian date – Calendar date)

The easiest Julian Date converter ever.

To convert a date/time to Julian; Simply enter the time (optional) and date and do **XEQ'CJC** and you will get the Julian Date back as a result. Press R/S and you will get the Julian Time of the Julian Date. Press R/S again and you will get the Julian Date plus the Julian Time (the total)

To convert a Julian Date to calendar date; Simply enter the Julian Date (with decimals if you want to cater for the time of the date as well) and do **XEQ'CJC** and you will get the calendar date in X and the time of day in Y.

The CJC program converts back and forth and back and forth and back and...

The program recognizes the input to discern if it is a Julian Date number or a Calendar Date number.

Simple.