СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Когда в 1992 году была выпущена в свет книга “Cartes à puce, Initiation et applications”[1], вряд ли кто-либо предполагал, что спустя три года невинные опыты в области чтения и записи пустых телефонных карт приведут к проникновению в секреты настоящих «электронных крепостей», каковыми могут считаться карты на микропроцессорах и новые телефонные карты с памятью, данные в которых можно обновлять.

Изготовители и дистрибьюторы самих карт, уверенные в эффективности механизмов защиты их приложений, оказали автору любезное содействие в написании данной книги, и теперь появилась возможность показать читателям, как можно преобразовать ПК в мощное орудие исследования и даже имитации самых разных чип-карт.

Несколько простейших электронных схем и специальное программное обеспечение, представленное на сервере www.dmk.ru издательства «ДМК», дадут вам материал для захватывающих открытий. Счастливого пути в увлекательный мир чип-карт!

---

[1] Русский перевод: «Чип-карты. Устройство и применение в практических конструкциях», М.: ДМК, 2000. Файлы с программным обеспечением размещены на сайте www.dmk.ru.

МИКРОПРОЦЕССОРЫ, МИКРОКАЛЬКУЛЯТОРЫ ИЛИ МИКРОКОНТРОЛЛЕРЫ?

В 1981 году, за два года до внедрения первых чип-карт, у компании BULL появились первые однокристалльные чип-карты на микрокалькуляторах.

По тем временам это было большое открытие, достигнутое благодаря сотрудничеству с компанией Motorola, поскольку с момента возникновения первых прототипов, выпущенных в 1974 году для компании Roland Moreno, имело место сочетание по крайней мере двух различных чипов (микропроцессора и памяти).

В 1976 г. компания CII Honeywell-Bull приобрела лицензию у фирмы Innovatron и приняла термин «карта на микрокалькуляторе» для обозначения продукта, который до сегодняшнего дня составляет семейство CP8. Хотя конкуренты BULL CP8 чаще употребляют сочетание «карта на микропроцессоре», в наибольшей степени соответствует истине название «карта на микроконтроллере».

На самом деле архитектура современных чипов для карт объединяет центральный процессор на 8 бит, ПЗУ, ОЗУ, ППЗУ и/или ЭСППЗУ, порты ввода-вывода точно так же, как и все микроконтроллеры. Просто в настоящее время существует тенденция добавлять все больше так называемых ресурсов для обеспечения безопасности, размещаемых на том же чипе. Таковы, вероятно, перспективы развития чип-карт.

МАТЕРИАЛЬНЫЕ РЕСУРСЫ

Если очевидным лидером на рынке инкартируемых микроконтроллеров является компания Motorola, то среди прочих производителей, по достоинству оценивших этот бизнес, наблюдается ожесточенная конкуренция. Карты начинают выпускаться в астрономических количествах!

Компания SGS-Thomson занимает особенно удачные позиции со своей продукцией, основанной на ядре, совместимом с процессорами Motorola, и очень интересными решениями в области защиты. Philips решительно выходит на сцену с архитектурой 8051, внося существенный вклад в использование возможностей криптографических вычислений. Компания Texas Instruments, прямой конкурент SGS-Thomson на рынке чипов для телефонных карт, предлагает, со своей стороны, ряд инкартируемых микропроцессоров, выросших из семейства TMS370. И конечно, данным вопросом очень пристально интересуются японские производители, в первую очередь концерн OKI с его продукцией OSCAR.

Интересно в общих чертах сравнить блок-схемы микроконтроллеров различных марок, часто встречающихся в чип-картах. На рис. 1.1 воспроизводедена общая структурная схема всех продуктов Motorola, которые имеют в своей основе классическое ядро 68HC05.

Версией SC24 (3K ПЗУ, 1K ЭСППЗУ, 128 байт ОЗУ) снабжены, в частности, французские банковские карты.

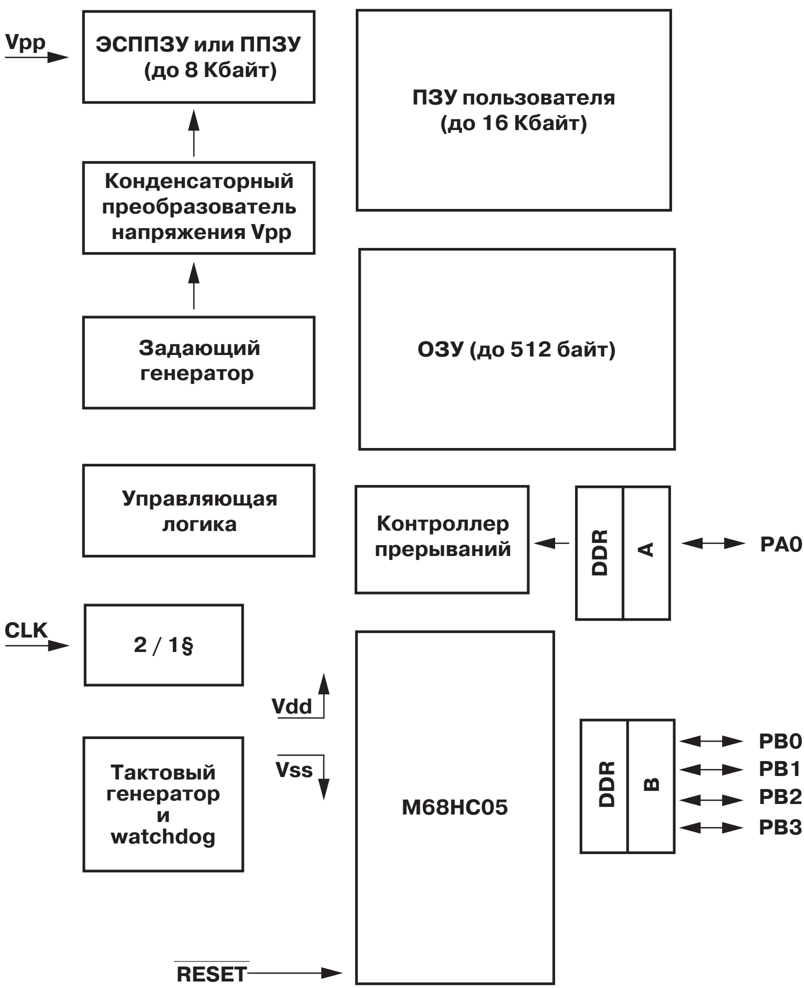

Рис. 1.1. Архитектура M68HC05SC Motorola

На рис. 1.2 показана архитектура кристаллов ST16XYZ, которая является «коньком» SGS-Thomson. Она объединяет совместимый с 6805 центральный процессор, наделенный рядом защитных функций, как аппаратных, так и программных: логическим блоком, который обеспечивает безопасность благодаря *датчикам*, определяющим попытки незаконного доступа к чипу; ПЗУ с *шифрованной шиной адреса*; системой маскировки колебаний тока питания при изменениях содержимого памяти; матрицей доступа, ограничивающей возможность несанкционированного доступа к памяти и т.д.

Отметим, что версией ST16301B на конкурентной основе снабжены французские банковские карты.

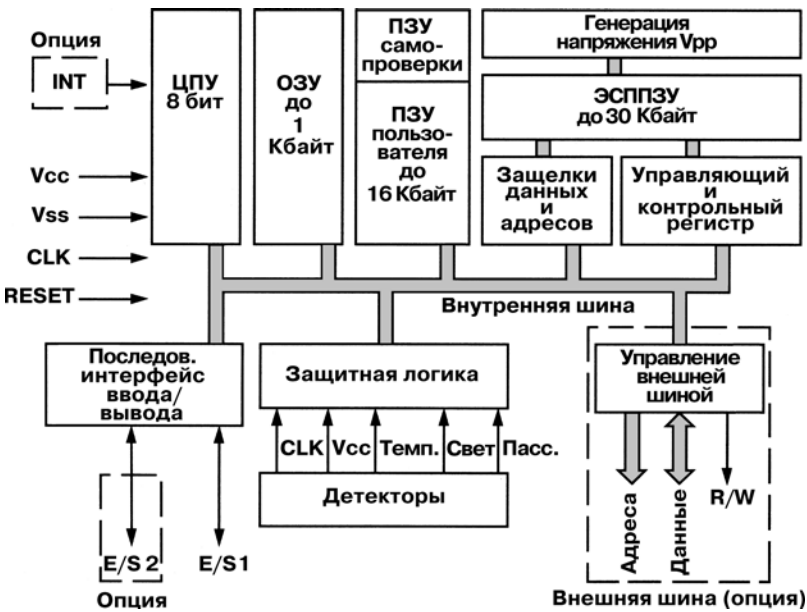

Рис. 1.2. Архитектура ST16XYZ SGS-Thomson

На рис. 1.3 продемонстрирован чрезвычайно прогрессивный процессор Philips, 83C852. Его ядру типа 8051 оказывает содействие специализированная *пересчетная система*, которая представляет собой настоящий быстродействующий сопроцессор, предназначенный для специальных операций, необходимых для кодирования данных. Все это содержится на одном чипе.

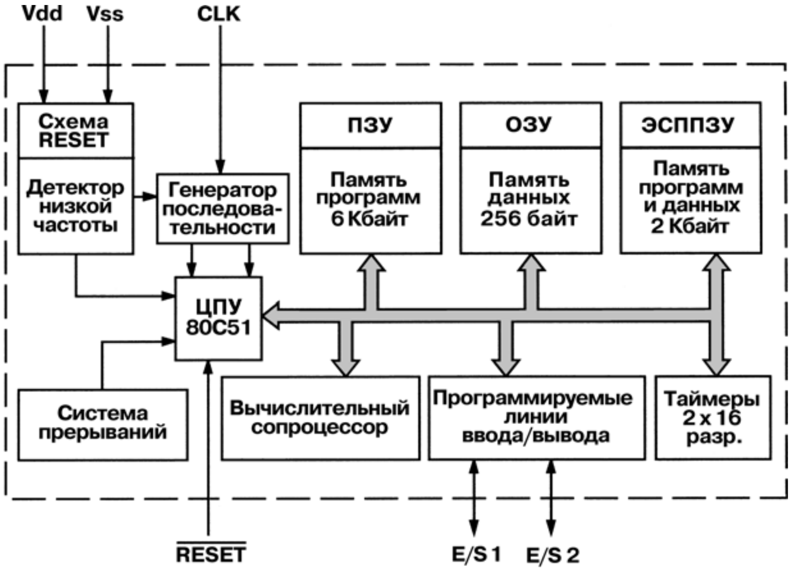

Рис. 1.3. Архитектура 83C852 Philips

Наконец, рис. 1.4 показывает внутреннюю организацию новейшего, но уже ставшего классическим чипа Texas Instruments – TMS373C007. Некоторые серии французских банковских карт снабжены чипом более поздней серии TMS373C012.

Можно отметить, что базовая структура карт разных марок почти не изменяется. Практически все разновидности (компоненты) располагают, например, двумя линиями последовательного ввода/вывода, хотя в настоящее время почти всегда используется только одна (*half-duplex*).

Ресурсы памяти, часто изменяемые в соответствии с требованиями клиента, практически всегда зависимы от одних и тех же технологических ограничений, накладываемых площадью кремния, которую можно удобно расположить на карте (приблизительно 4×6 мм). Правда, применение субмикронных технологий дает шанс улучшить ситуацию в относительно скором будущем.

К тому же многое можно расположить на 8 Кбайт ЭСППЗУ (перезаписываемого) или ППЗУ (применяемого без возможности повторного использования), а в 16 Кбайт ПЗУ удастся разместить код вполне достойной *операционной системы*.

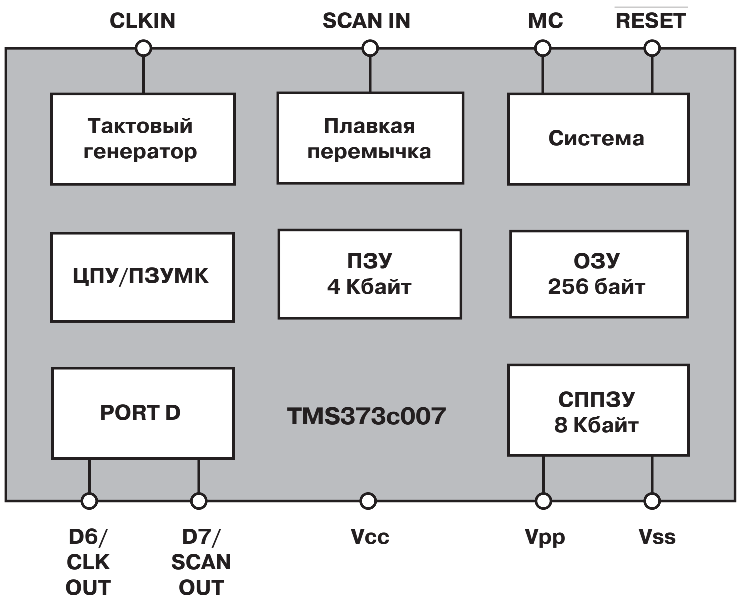

Рис. 1.4. Архитектура TMS373C007

ПРОГРАММНЫЕ РЕСУРСЫ

Все чипы карт на микроконтроллерах происходят напрямую от того или иного классического семейства микропроцессоров или микроконтроллеров. От последних они наследуют бóльшую часть системы команд, даже если принято удалять из них несколько функций, не представляющих особого интереса в области чип-карт; правда, вместо таких команд добавляются более специфические.

Компания Texas Instruments разработала команду NOVP (NOP с варьируемой длиной, очень полезную для выполнения операции выдержки времени), а также команду перемещения бит MOVB.

В табл. 1.1 воспроизводится полная система команд, наиболее широко распространенная в мире карт на микропроцессорах, так как она в значительной степени является общей для чипов SGS-Thomson и Motorola.

В принципе основные манипуляции с картами на микропроцессорах не нуждаются в прямом программировании центрального процессора (и даже не позволяют этого). С точки зрения удобства использования,

Таблица 1.1. Система команд для чип-карт ST16XYZ

Register/Memory and Absolute Jump Group –
группа Регистр/память и Безусловные переходы

Function	**Mnemonic**	**Addressing Mode**					
		Immediate	**Direct**	**Extended**	**Index no offset**	**Index 8 bit offset**	**Index 16 bit offset**
Load A with memory	LDA	$_{2}$A6^{2}	$_{2}$B6^{3}	$_{3}$C6^{4}	$_{1}$F6^{3}	$_{2}$E6^{4}	$_{3}$D6^{5}
Load X with memory	LDX	$_{2}$AE^{2}	$_{2}$BE^{3}	$_{3}$CE^{4}	$_{1}$FE^{3}	$_{2}$EE^{4}	$_{3}$DE^{5}
Load memory with A	STA		$_{2}$B7^{3}	$_{3}$C7^{4}	$_{1}$F7^{3}	$_{2}$E7^{4}	$_{3}$D7^{5}
Load memory with X	STX		$_{2}$BF^{3}	$_{3}$CF^{4}	$_{1}$FF^{3}	$_{2}$EF^{4}	$_{3}$DF^{5}
Add memory to A	ADD	$_{2}$AB^{2}	$_{2}$BB^{3}	$_{3}$CB^{4}	$_{1}$FB^{3}	$_{2}$EB^{4}	$_{3}$DB^{5}
Add memory and carry to A	ADC	$_{2}$A9^{2}	$_{2}$B9^{3}	$_{3}$C9^{4}	$_{1}$F9^{3}	$_{2}$E9^{4}	$_{3}$D9^{5}
Subtract memory to A	SUB	$_{2}$A0^{2}	$_{2}$B0^{3}	$_{3}$CO^{4}	$_{1}$F0^{3}	$_{2}$E0^{4}	$_{3}$D0^{5}
Subtract memory with carry	see	$_{2}$A2^{2}	$_{2}$B2^{3}	$_{3}$C2^{4}	$_{1}$F2^{3}	$_{2}$E2^{4}	$_{3}$D2^{5}
And memory to A	AND	$_{2}$A4^{2}	$_{2}$84^{3}	$_{3}$C4^{4}	$_{1}$F4^{3}	$_{2}$E4^{4}	$_{3}$D4^{5}
Or memory with A	ORA	$_{2}$AA^{2}	$_{2}$BA^{3}	$_{3}$CA^{4}	$_{1}$FA^{3}	$_{2}$EA^{4}	$_{3}$DA^{5}
Exclusive OR	EOR	$_{2}$A8^{2}	$_{2}$B8^{3}	$_{3}$C8^{4}	$_{1}$F8^{3}	$_{2}$E8^{4}	$_{3}$D8^{5}
Arithmetic Compare A	CMP	$_{2}$A1^{2}	$_{2}$B1^{3}	$_{3}$C1^{4}	$_{1}$F1^{3}	$_{2}$E1^{4}	$_{3}$DI^{5}
Arithmetic Compare X	CPX	$_{2}$A3^{2}	$_{2}$B3^{3}	$_{3}$C3^{4}	$_{1}$F3^{3}	$_{2}$E3^{4}	$_{3}$D3^{5}
Bit compare A and memory	BIT	$_{2}$A5^{2}	$_{2}$B5^{3}	$_{3}$C5^{4}	$_{1}$F5^{3}	$_{2}$E5^{4}	$_{3}$D5^{5}
Absolute Jump	JMP		$_{2}$BC^{2}	$_{3}$CC^{3}	$_{1}$FC^{2}	$_{2}$EC^{3}	$_{3}$DC^{4}
Jump to subroutine	JSR		$_{2}$BD^{5}	$_{3}$CD^{6}	$_{1}$FD^{5}	$_{2}$ED^{6}	$_{3}$DD^{7}

Bit manipulation and test Group – группа обработки и тестирования битов

Function	**Mnemonic**	**Addressing Mode**
Bit Set	BSET b (b=0..7)	$_{2}$(10+2*b)^{5}
Bit clear	BCLR b (b=0..7)	$_{2}$(11+2*b)^{5}
Test bit b and branch if true	BRSET b (b=0..7)	$_{3}$(00+2*b)^{5}
Test bit b and branch if false	BRCLR b (b=0..7)	$_{3}$(01+2*b)^{5}

Таблица 1.1. Система команд для чип-карт ST16XYZ (продолжение)

Read/Modify/Write Group – группа Чтение/Изменение/Запись

Function	Mnemonic	Addressing Mode				
		Inherent A	Inherent X	Direct	Index no offset	Index 8 bit offset
Increment	INC	$_1$4C^2	$_1$5C^2	$_2$3C^5	$_1$7C^5	$_2$6C^6
Decrement	DEC	$_1$4A^2	$_1$5A^2	$_2$3A^5	$_1$7A^5	$_2$6A^6
Clear	CLR	$_1$4F^2	$_1$5F^2	$_2$3F^5	$_1$7F^5	$_2$6F^6
One's Complement	COM	$_1$43^2	$_1$53^2	$_2$33^5	$_1$73^5	$_2$63^6
Negate (2's Complement)	NEG	$_1$40^2	$_1$50^2	$_2$30^5	$_1$70^5	$_2$60^6
Rotate Left thru Carry	ROL	$_1$49^2	$_1$59^2	$_2$39^5	$_1$79^5	$_2$69^6
Rotate Right thru Carry	ROR	$_1$46^2	$_1$56^2	$_2$36^5	$_1$76^5	$_2$66^6
Logical Shift Left into Carry	LSL	$_1$48^2	$_1$58^2	$_2$38^5	$_1$78^5	$_2$68^6
Logical Shift Right into Carry	LSR	$_1$44^2	$_1$54^2	$_2$34^5	$_1$74^5	$_2$64^6
Arithmetic Shift Right into Carry	ASR	$_1$47^2	$_1$57^2	$_2$37^5	$_1$77^5	$_2$67^6
Test for Negative or Zero	TST	$_1$4D^2	$_1$5D^2	$_2$3D^3	$_1$7D^3	$_2$6D^4

Branch Group – группа ветвления (условных переходов)

Function	Mnemonic	Addressing Mode
		RELATIVE
Branch Always	BRA	$_2$20^3
Branch Never	BRN	$_2$21^3
Branch if Higher	BHI	$_2$22^3
Branch if Unsigned Lower or Same	BLS	$_2$23^3
Branch if Carry Clear	BCC	$_2$24^3
Branch if Unsigned Higher or Same	BHS	$_2$24^3
Branch if Carry Set	BCS	$_2$25^3
Branch if Unsigned Lower than	BLO	$_2$25^3
Branch if Not Equal	BNE	$_2$26^3
Branch if Equal	BEQ	$_2$27^3
Branch if Half Carry Clear	BHCC	$_2$28^3
Branch if Not Half Carry Set	BHCS	$_2$29^3
Branch if Plus	BPL	$_2$2A^3
Branch if Minus	BMI	$_2$2B^3
Branch if Not Interrupt Mask	BMC	$_2$2C^3
Branch ff Interrupt Mask	BMS	$_2$2D^3
Branch if Interrupt Line Low	BIL	$_2$2E^3
Branch if Interrupt Line High	BIH	$_2$2F^3
Branch to Subroutine	BSR	$_2$AD^5

Таблица 1.1. Система команд для чип-карт ST16XYZ (окончание)

Miscellaneous Group – дополнительная группа

Function	Mnemonic	Addressing Mode
		INHERENT
Multiply (X : A=X* A)	MUL	$_{1}42^{10}$
Transfer A to X	TAX	$_{1}97^{2}$
Transfer X to A	TXA	$_{1}9F^{2}$
Transfer SP to A	TSA	$_{1}9E^{2}$
Clear Carry Flag	CLC	$_{1}98^{1}$
Set Carry Flag	SEC	$_{1}99^{1}$
Clear Interrupt Mask bit	CLI	$_{1}9A^{2}$
Set Interrupt Mask bit	SEI	$_{1}9B^{2}$
Reset Stack Pointer	RSP	$_{1}9C^{2}$
No Operation	NOP	$_{1}90^{2}$
Return from Interrupt	RTI	$_{1}80^{2}$
Return from Subroutine	RTS	$_{1}81^{s}$
Software Interrupt	SWI	$_{1}83^{9}$
Halt CPU/Enable INT	WAIT	$_{1}8F^{2}$
Halt CPU/STOP Clocks/Enable INT	STOP	$_{1}8E^{2}$

а также и безопасности, процессор полностью находится под контролем операционной системы, расположенной в ПЗУ.

Есть, впрочем, исключение, лишь подтверждающее правило: некоторые карты, так называемые COS, обладают своеобразным *портом ввода*, позволяющим «имплантировать» некоторый объем собственного кода в ППЗУ или в ЭСППЗУ. В этом случае необходимо хорошее знание системы команд процессора. Именно эти карты часто снабжены чипом ST16CXYZ.

РЕСУРСЫ, ОБЕСПЕЧИВАЮЩИЕ БЕЗОПАСНОСТЬ

Каким бы ни было разрабатываемое приложение, назначение чип-карты можно всегда свести к сохранению данных в более или менее защищенной форме – ведь речь идет о картах с памятью.

Превосходство карты на микропроцессоре по отношению к обычной карте с памятью, даже защищенной, обусловлено, с одной стороны, более простым и упорядоченным протоколом связи, а с другой – способностью проводить сложную обработку данных, которые карта принимает или передает.

Благодаря *криптографическим ресурсам* карта на микропроцессоре может не дать двух идентичных ответов на один и тот же запрос, по крайней мере, если правильно использовать возможности, предлагаемые ее операционной системой. Очевидно, что подобная организация в высшей степени усложняет всякую попытку *повторного воздействия*, то есть перехвата и последующего воспроизведения ответа карты.

Однако, даже если принять во внимание вышесказанное, некоторые секретные сведения, содержащиеся в памяти, ни в коем случае не должны покидать карту: в идеале только операционная система имеет возможность доступа к ним, используя их в качестве элементов расчетов. Притом в открытом виде может появиться только результат. Подобным образом действуют *конфиденциальные коды*, которые присваиваются владельцу карты (*транспортный код, ключ владельца*) или ее дистрибьютору: нет никакой возможности прочитать их, как с простой магнитной полосы, чтобы сравнить вне карты с кодом, набранным на клавиатуре!

Процедура обеспечения безопасности состоит в том, чтобы предъявить код карте, которая будет проверять его с помощью своей операционной системы, внутри. После проведения такой проверки карта может информировать (открыто или закодированным способом) о том, что код правилен или ложен, или довольствоваться выдачей разрешения на доступ в зону памяти, которая до этого была закрыта. Столь высокий уровень безопасности позволяет использовать карты или ключи на микропроцессоре в финансовой сфере (банковские карты, электронные кошельки и т.д.), области сотовой связи (телефоны GSM) или платного телевидения.

На рис. 1.5 показывается (согласно идее SGS-Thomson), как построены телевизионные декодеры, совместимые с наиболее современными кодирующими системами.

Снабженная памятью ЭСППЗУ с содержимым, подлежащим многократным изменениям, карта содержит все секретные данные, которые, наряду с получаемыми во время передачи, необходимы микропроцессору декодера для управления своим цифровым декодирующим устройством. Информация, содержащаяся в карте, может изменяться дистанционно с помощью обычного телевизионного передатчика посредством команд, включаемых в видеосигнал. В случае глубоких изменений в системе намного практичнее рассылать по почте новые карты, чем заниматься обменом декодеров.

Интересно отметить, что чипы, используемые в этой системе, являются гораздо более мощными, чем применяемые в банковских картах!

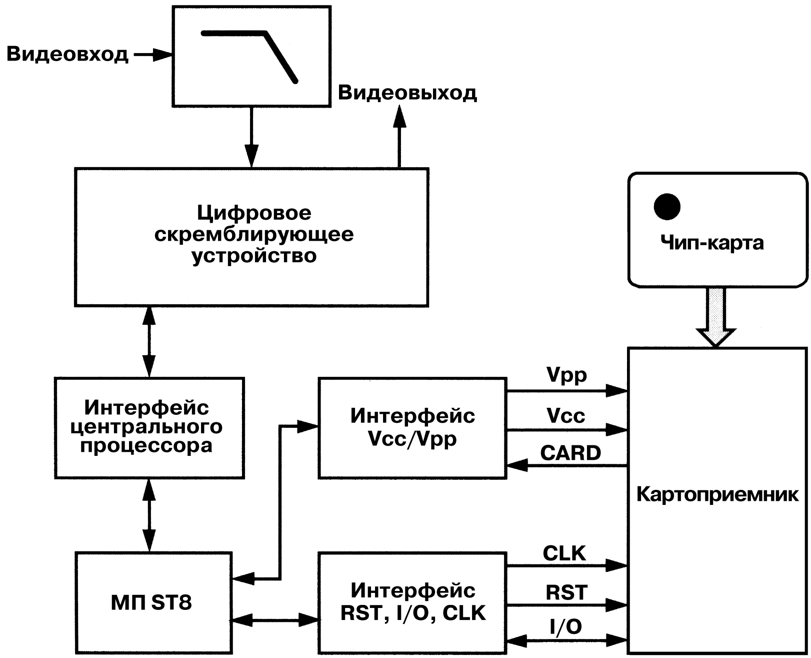

Рис. 1.5. Блок-схема телевизионного декодера с картой

НЕСКОЛЬКО ПОНЯТИЙ ИЗ КРИПТОГРАФИИ

Цель кодирования (науки составления секретных сообщений) заключается в возможности передачи конфиденциальных сведений в форме сообщения, лишенного какого-либо видимого смысла, на случай, если кто-нибудь перехватит послание. Для этого приступают к шифрованию данных с помощью ключа, затем к их дешифрованию посредством того же самого ключа (так называемый симметричный алгоритм) или другого (асимметричный алгоритм).

Очевидно, что если ключ раскодирования идентичен ключу кодирования, его защита предполагает те же проблемы, что и защита данных. На самом деле можно, конечно, расшифровать сообщение, если знать ключ и алгоритм кодирования. Но это вероятно и при полном незнании алгоритма, если имеются ключ и образцы некодированных сообщений в сопровождении их кодированной версии.

Одна из наиболее простых процедур шифрования в информатике – проведение операции Исключающее ИЛИ между каждым байтом кодируемых данных и каждым байтом ключа. Данный метод интересен постольку, поскольку один и тот же алгоритм, применяемый с одинаковым ключом, позволяет проведение операции декодирования.

При условии безупречной защиты ключа и его однократного использования даже такая простейшая процедура обеспечивает абсолютную защиту в случае, если длина ключа (число байтов) будет, по крайней мере, равна длине кодируемого сообщения. Если напротив, один и тот же ключ небольшой длины неоднократно используется для кодирования различных частей гораздо более длинного сообщения, это ставит под сомнение всю надежность шифровки.

Небольшая программа `XOR.BAS` позволяет познакомиться с этим элементарным методом кодирования, правда, в упрощенном виде: ключ состоит из единственного байта, то есть одна буква всегда выражается одним и тем же числом (это соответствует шифрованию замещением, аналогичному той системе тайнописи, которую используют скауты). Нельзя не согласиться, что если длина сообщения ограничивается одним байтом, то совершенно невозможно обойтись без ключа для его раскодирования.

```
10 REM -- XOR.BAS --
20 A$="(c)1995 Патрик Гелль"
30 FOR G=1 TO LEN(A$)
40 D$=MID$(A$,G,1):D=ASC(D$)
50 R=D XOR 47
55 REM общий ключ = 47
70 PRINT D$,R,
80 K=R XOR 47
100 PRINT CHR$(K)
110 NEXT G
120 END
```

Проблема ключей была очень искусно решена по так называемому принципу алгоритмов на открытых ключах. Подобная криптографическая система использует два различных ключа: один для кодирования и другой для декодирования. В общем случае ключ кодирования является *открытым*, а декодирования – *секретным*: для того, чтобы отправить кому-либо сообщение, у адресата узнают открытый код (в крайнем случае его разыскивают в специальном справочнике).

Даже если алгоритмы кодирования и декодирования являются открытыми, расшифровать закодированные сообщения может только лицо, обладающее секретным кодом, с помощью своего открытого ключа. Но система работает и по обратному принципу: сообщение, зашифрованное с помощью секретного ключа, может быть расшифровано любым владельцем соответствующего открытого ключа, который таким образом будет иметь формальное доказательство происхождения сообщения (речь идет об электронной подписи).

Наиболее известным алгоритмом на открытом ключе является алгоритм RSA (Rivest Shamir Adleman), который использует математические свойства степеней по модулю N: выбираются два простых числа (p и q), произведение которых (pq) послужит модулем для вычисления последующих степеней. Проще говоря, A в степени B по модулю N представляет собой A, умноженное B раз на A, минус столько раз N, чтобы получить положительный результат, меньший чем модуль N.

Алгоритм RSA основывается на том факте, что

(p – 1)(q – 1)/x = 1 по модулю pq

при условии, что x не делится ни на p, ни на q.

На практике это условие выполняется автоматически, если каждое из чисел *p* и *q* больше *x*, что, в общем, характерно при кодировании (чаще всего работают на числах разрядностью 512 бит).

```
10 REM -- MODULO.BAS --
20 KEY OFF:CLS
30 INPUT"Данные? ",D
40 INPUT"Показатель степени? ",E
50 INPUT"Коэффициент? ",M
60 R=D
70 FOR F=1 TO E-1
80 R=R*D
90 IF R<M THEN 110
100 R=R-M:GOTO 90
110 NEXT F
120 PRINT"результат: ";R
130 PRINT:GOTO 30
140 REM (c)1995 Patrick GUEULLE
```

Небольшая программа `MODULO.BAS` позволит провести несколько экспериментов со свободно выбранными операндами, а кроме этого проиллюстрирует принцип, используемый для возведения в степень по модулю с помощью программы на языке BASIC, без особого риска возникновения ошибок переполнения (overflow).

Оба ключа RSA (открытый e и секретный d или наоборот) выбираются так, чтобы они отвечали условию:

ed = 1 модуль(p – 1)(q – 1).

В таком случае мы имеем:

$(x^d)^e$ = x(модуль pq)

При этом ничто не позволяет вывести d из e или наоборот.

```
10 REM -- RSA.BAS --
20 A$="(c)1995 Патрик Гелль"
30 FOR G=1 TO LEN(A$)
40  D$=MID$(A$,G,1):D=ASC(D$)
50 E=15:M=391
55 REM Открытый ключ = 15, коэффициент = 391
60 GOSUB 130
70 PRINT D$,R,
80 E=47:M=391
85 REM Секретный ключ = 47, коэффициент = 391
90 D=R:GOSUB 130
100 PRINT CHR$(R)
110 NEXT G
120 END
130 R=D
140 FOR F=1 TO E-1
150 R=R*D
160 IF R<M THEN 180
170 R=R-M:GOTO 160
180 NEXT F
190 RETURN
```

Программа RSA.BAS применяет данный алгоритм в условиях, полностью сравнимых с вышеназванными, при использовании следующих ключей:

- открытый ключ: 15, модуль 391;
- секретный ключ: 47, модуль 391.

Значения получены из простых чисел p = 17 и q = 23.

Конечно, надежность операции не выше, чем в предыдущем случае, так как обработка идет байт за байтом, в то время как лучше было бы обрабатывать блоки по 512 бит. Однако преимущество этого способа в том, что он показывает, насколько замедлены вычисления даже при работе с блоками по 8 бит: именно благодаря этому ныне возможно снабжать чип-карты алгоритмом RSA только при наличии на карте или на чипе специального сопроцессора, пока не появились «крутые» чип-карты с 32-битным ядром. Например, с микроконтроллером 83C852 удается провести вычисления за полторы секунды, в то время как для проведения аналогичной операции с помощью базовой системы команд микропроцессора на 8 бит понадобилось бы почти три минуты. По этой причине большинство карт на микропроцессорах довольствуется симметричным алгоритмом, ультрасекретный ключ которого физически присутствует на карте и в связи с этим должен очень тщательно защищаться.