распознаваемых модулем распознавания мелодии. Во-вторых, система может с небольшими преобразованиями быть использованной для определения музыкального плагиата.

Ранее пакет HTK использовался для распознавания речи [2]. Релевантность распознавания слов достигала 80%. В нашем случае, этот процент более низкий.

Такой процент распознавания обусловлен посторонними шумами и характеристиками звукозаписывающей аппаратуры.

Литература

1. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Rabiner, L.* б.м.: IEEE Press, 1988.
2. The HTK Book (for HTK Version 3.4).*Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J. Ollason, D., Valtchev, V., Woodland, P.* 2006.
3. *Baldridge Jason.* Lexically Specified Derivational Control in Combinatory Categorial Grammar. Edinburgh: б.н., 2002.
4. *Baum L.*: An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology/ pp.360-363. Bull Amer. Meteoral Sac. 1969.
5. *Baum L.* Statistical inference for probabilistic functions of nite state Markov chains / Baum L. E., Petrie T., Ann. Math. Stat. Vol. 37. P. 1554–1563. 1960.
6. *Viterbi A.* Error bounds for convolutional codes and an asymptotically optimal decoding algorithm / IEEE Trans. Informat. Theory. P. 260–269. 1967.

РАЗРАБОТКА СИСТЕМЫ ШИФРАЦИИ НА БАЗЕ RISC МИКРОКОНТРОЛЛЕРОВ NEC V850

А. В. Пильгун, П. П. Коржуков

ВВЕДЕНИЕ

Для предотвращения несанкционированного доступа уже давно используются криптографически защищенные каналы связи. С этой целью был разработан ряд практических систем шифрации. Эти системы тщательно протестированы и гарантируют потребителю заданную криптостойкость. Системы открытые, опубликованы, имеют своих последователей и доступные программные и аппаратные реализации.

Однако эти системы являются универсальными и в каждом конкретном случае их необходимо адаптировать для уменьшения стоимости и повышения эффективности.

Вычислительные ресурсы даже простого человека достигли значительных высот, что делает возможным совершать все более крупные атаки. Это подталкивает совершенствовать и технические методы защи-

ты, что говорит об актуальности данного исследования. Часто выделенные аппаратные устройства шифрования становятся непреодолимым барьером для злоумышленника, т.к. к ним нет доступа извне.

Современные 32-разрядные микроконтроллеры являются достаточно мощными вычислителями и используются повсеместно. Устройства шифрования, построенные на них, востребованы как в коммерческой сфере, так и среди обыкновенных пользователей, например, в рамках концепции «Умный дом». Проект «Умный дом» предполагает обмен данными внутри себя и с внешними устройствами. Передаваемые данные могут иметь не конфиденциальный характер, но при попадании в руки злоумышленника нанести пользователю вред. Поэтому следует использовать шифрование данных при передачи по незащищенным каналам.

ОПТИМИЗАЦИЯ АЛГОРИТМОВ ШИФРАЦИИ ПОД КОНКРЕТНЫЙ ВЫЧИСЛИТЕЛЬ

Алгоритмы могут быть оптимизированы по размеру и по коду. Эти стратегии находятся между собой в отношении антагонизма. Часто необходимо получить максимально быстрое устройство шифрования, так как криптографические преобразования занимают время и могут наложить ограничение на скорость передачи данных. Однако рассматриваемая платформа имеет скромные ресурсы, как по скорости работы, так и по размеру памяти. Это делает необходимым рассмотреть обе стратегии.

Оптимизация проводилась на примере системы шифрации DES, реализация которой предложена в [1]. Данная реализация является универсальной и написана на языке С. Для получения эффективной реализации необходимо оптимизировать этот алгоритм в соответствии с архитектурой аппаратной платформы. На примере систем шифрации ГОСТ 28147-89, AES 128, DES исследовались возможности компилятора IAR System оптимизировать код.

При исследовании системы шифрации DES был произведен подсчет количество тактов функции генерации подключей «deskey» и функций зашифровывания и расшифровывания. В результате оказалось, что функция генерации подключей занимает больше всего тактов микроконтроллера. На ней и осуществляется оптимизация.

В процессе оптимизации функции «deskey» было обнаружено, что при обращении к массивам типа long, занимающих в данной системе 4 байта, индексным способом компилятор генерирует лишние команды для интерпретации указателя на память с помощью индекса. Переход к

указателям избавляет от использования переменной инкрементирования в цикле for при обращении по индексу массива (табл. 1).

Таблица 1

Генерирование ассемблерного кода компилятором

Индексная запись		Запись через указатели	
if(pcr[pc2[j]])	kn[m] |= bigbyte[j];	if(pcr[pc2[j]])	*pknm |= *pbigbyte
LD.BU	pc2[r26],r1	LD.BU	pc2[r26],r1
ZXB	r1	ZXB	r1
ADD	sp,r1	ADD	sp,r1
LD.BU	0[r1],r1	LD.BU	0[r1],r1
ZXB	r1	ZXB	r1
CMP	0,r1	CMP	0,r1
BE	??deskey_18	BE	??deskey_18
MOV	**r23,r5**	LD.W	0[r22],r1
SHL	**2,r5**	LD.W	0[r20],r5
ADD	**sp,r5**	OR	r1,r5
MOV	**r23,r1**	ST.W	r5,0[r22]
SHL	**2,r1**		
ADD	**sp,r1**		
LD.W	**56[r1],r1**		
MOV	**r26,r6**		
SHL	**2,r6**		
LD.W	**bigbyte[r6],r6**		
OR	r1,r6		
ST.W	r6,56[r5]		

При использовании циклов с указателями отпадает необходимость в использовании инкрементируемой в цикле переменной. Это дает возможность использовать цикл while и, тем самым сократить каждую итерацию ещё на одну операцию: использование цикла вида while (pc1j <= pc1je) вместо цикла for (j = 0; j < 56; j++), где pc1j и pc1je – указатели на начало и конец массива соответственно.

В исходной реализации в цикле использовались временные переменные. Отказываясь от временных переменных, путем вычисления значений по мере необходимости можно дополнительно уменьшить количество команд.

При выполнении работы были выявлены следующие моменты, которые необходимо использовать для оптимизации при реализации алгоритмов на маломощных вычислителях:

- объявлять небольшие глобальные константы с типом char. Таким образом, константы занимают меньший объем памяти;
- объявлять локальные переменные с типом int;
- использование указателей при обращении к массивам;

- использование при работе с указателями операций характерных для них (например, использование цикла while вместо for);
- избавление от временных переменных внутри цикла.

СРАВНЕНИЕ РЕАЛИЗАЦИЙ

Исходя из двух конкурирующих стратегий – по коду и по скорости, критериями оптимальности служат количество тактов и количество сгенерированного кода и данных. Количество тактов было посчитано при помощи регистра CYCLECOUNTER в режиме отладки в среде разработки IAR Systems. Данные о количестве сгенерированного кода брались из map-файла сгенерированного при линковке программы.

По результатам исследования были получены характеристики систем шифрации при реализации на конкретной архитектуре ядра V850ES, что позволяет сравнить их для данной платформы.

Таблица 2

Количество тактов

Алгоритмы	Инициализация и генерация ключевой информации	Зашифровывание	Расшифровывание	Длина ключа
DES	52 773	1149	1149	56
AES 128	116210	1965	1947	128
ГОСТ 28147-89	12187+99	1265	1266	256

Алгоритмы DES и ГОСТ часто сравниваются из-за своей похожести [2]. По результатам табл. 2 и 3 действительно можно сказать, что их реализации конкурируют. Количество тактов, выполняемых в процессе шифрования сравнимы, как и объем занимаемой памяти. Если DES имеет небольшой выигрыш в скорости, то ГОСТ имеет так же небольшой выигрыш в размере.

Таблица 3

Размер сегментов, байт

Оптимизация	CODE	DATA	CONST
DES	3 354	4 224	2 368
AES	3 822	5 164	84
ГОСТ 28147-89	2 500	4 096	188

Исходя из табл. 2 реализация алгоритма шифрации DES является самой быстрой. Данная система является одной из самых известной, однако имеет уже значительный возраст и короткую длину ключа для достаточной защиты, что практически исключает возможность ее использования. Реализация алгоритма ГОСТ 28147-89 имеет характеристики близкие к DES, при этом имея длину ключа 256 битов. Кроме того, эта сис-

тема хорошо известна и распространена в странах СНГ, что еще больше упрощает ее использование на данной территории. Алгоритм AES имеет достойные характеристики, не исключено, что при грамотной реализации он может по скорости сравняться с DES и ГОСТ 28147-89.

Литература

1. *Шнайер Б.* Прикладная криптография. Протоколы, алгоритмы и исходные тексты на языке С / Шнайер Б. – 2-е издание.
2. *Винокуров А.* Сайт Андрея Винокурова http://www.enlight.ru/crypto/ frame.htm.

АНАЛИЗ РЕЖИМА АВТОПУЛЬСАЦИЙ ИЗЛУЧЕНИЯ В ПОЛУПРОВОДНИКОВОМ ИНЖЕКЦИОННОМ ЛАЗЕРЕ ПРИ ВНЕШНЕЙ ОПТИЧЕСКОЙ СИНХРОНИЗАЦИИ

С. Г. Савицкий

Внешняя оптическая синхронизация – это явление, при котором лазер генерирует на частоте вводимого в него излучения. Для достижения в данный лазер, называемый ведомым, инжектируется излучение другого лазера, называемого ведущим. При определенных значениях параметров вводимого излучения ведомый лазер синхронизируется и используется при этом как усилитель. Явление позволяет значительно улучшить динамические характеристики ведомого лазера, в частности, увеличить полосу модуляции, уменьшить относительную интенсивность шума. Учитывая, что стоимость полупроводниковых лазеров относительно невысокая, синхронизация внешним излучением становится весьма привлекательной для применений.

Данные системы также могут быть использованы при значениях параметров внешнего излучения, не приводящих к синхронизации ведомого лазера. Например, возможно получение режима с колебаниями выходной мощности. Это позволяет использовать данные системы в качестве генератора колебаний. В данной работе мы подробно рассмотрим, какие значения параметров приводят к подобным режимам.

Анализ медленных изменений амплитуды электромагнитного поля будем проводить на основе приближенных скоростных уравнений вместо использования точного волнового уравнения. Система уравнений для плотности носителей заряда n, плотности фотонов S и фазы излучения φ имеет вид [1; 2]:

$$\frac{\mathrm{d}n}{\mathrm{d}t} = \frac{j}{ed} - \frac{n}{\tau} - v_g G S, \tag{1}$$

$$\frac{\mathrm{d}S}{\mathrm{d}t} = v_g(G - k_l)S + 2v_g æ B\sqrt{S}\cos\varphi + \beta\frac{n}{\tau}, \tag{2}$$